



{K}ODE{K}LOUD

Vijin Palazhi

[@vijinpalazhi](#)



Objectives

Introduction
to IaC

Types of
IaC Tools

Why
Terraform?

HCL
Basics

Provision,
Update and
Destory

Providers

Input
Variables

Output
Variables

Resource
Attributes

Resource
Dependencies

Objectives

Terraform
State

Commands

Mutable vs
Immutable

LifeCycle
Rules

Datasources

Meta-
Arguments

count

for-each

Version
Constraints

AWS Basics

Objectives

Programmatic
Access

IAM Basics

IAM with
Terraform

Introduction
to S3

S3 with
Terraform

Introduction to
DynamoDB

DynamoDB
with
Terraform

Objectives

Remote
State

State Locking

Remote
Backend with
S3

State
Commands

Introduction
to EC2

AWS EC2
With
Terraform

Provisioners

Objectives

Terraform
Taints

Debugging

Terraform
Import

Modules

Functions

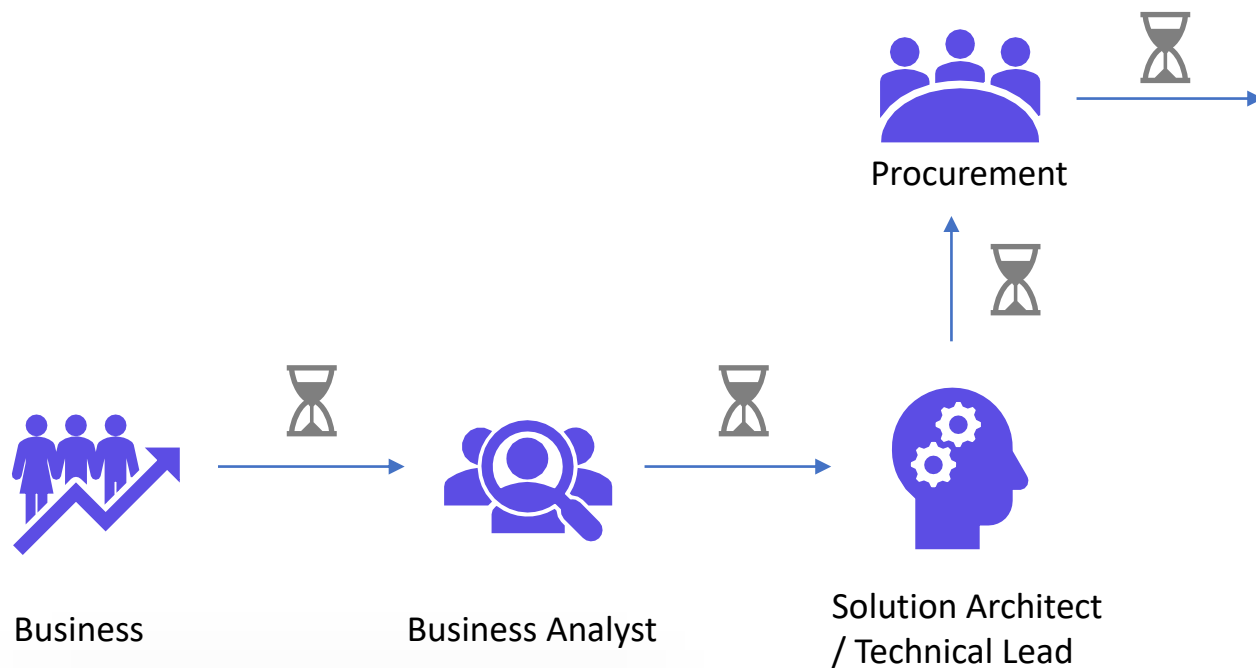
Conditional
Expressions

Workspaces

Terraform
Cloud

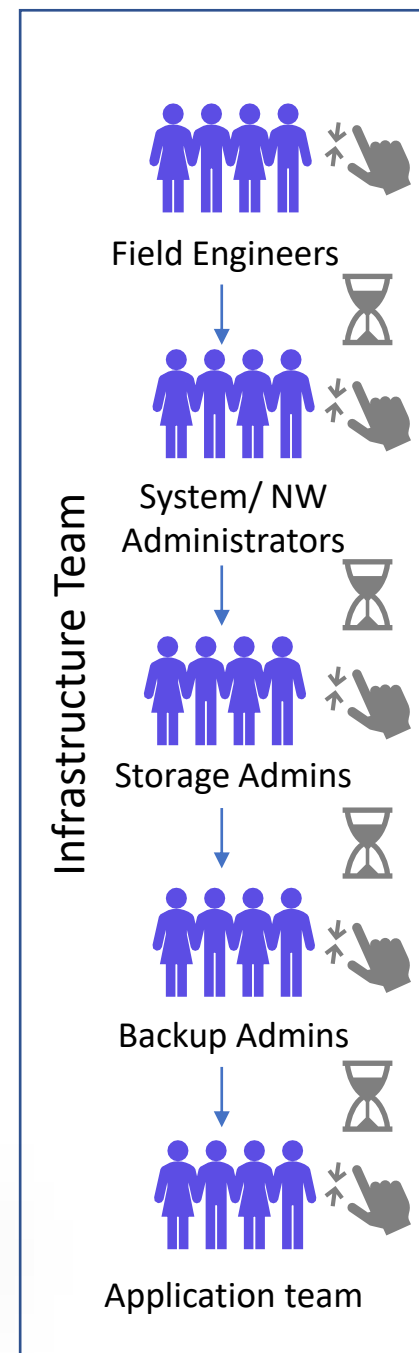


Traditional IT & Challenges



- Slow Deployment
- Expensive
- Limited Automation
- Human Error
- Wasted Resources

Inconsistency





Services ▾

Resource Groups ▾



1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Add Tags

6. Configure Security Group

7. Review

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

▼ AMI Details



Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0b1e2eeb33ce3d66f

**Free tier
eligible**

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extra

Root Device Type: ebs Virtualization type: hvm

▼ Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

▼ Security Groups

Security group name

launch-wizard-1

Description

launch-wizard-1 created 2020-07-09T15:48:36.426-04:00

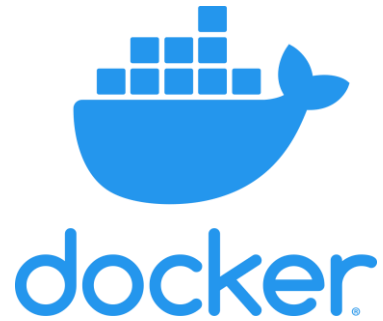
Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ	Description ⓘ
--------	------------	--------------	----------	---------------

This security group has no rules

▼ Instance Details


Number of instances 1**Purchasing option** On demand**Network** vpc-fe3baa86**Subnet** No preference (default subnet in any Availability Zone)

Infrastructure as Code





{K}ODE{K}LOUD

The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in a lighter shade of blue. Additionally, there are some geometric shapes in the corners: a light blue trapezoid in the top-left and a light blue parallelogram in the bottom-right.

Infrastructure as Code

Infrastructure as Code

ec2.sh

```
#!/bin/bash
```

```
IP_ADDRESS="10.2.2.1"
```

```
EC2_INSTANCE=$(ec2-run-instances --instance-type  
t2.micro ami-0edab43b6fa892279)
```

```
INSTANCE=$(echo ${EC2_INSTANCE} | sed 's/*INSTANCE //' |  
sed 's/ .*//')
```

```
# Wait for instance to be ready  
while ! ec2-describe-instances $INSTANCE | grep -q  
"running"  
do  
    echo Waiting for $INSTANCE is to be ready...  
done
```

```
# Check if instance is not provisioned and exit  
if [ ! $(ec2-describe-instances $INSTANCE | grep -q  
"running") ]; then  
    echo Instance $INSTANCE is stopped.  
    exit  
fi
```

```
ec2-associate-address $IP_ADDRESS -i $INSTANCE
```

```
echo Instance $INSTANCE was created successfully!!!
```



Services

Resource Groups



1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Add Tags

6. Configure Security Groups

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign your instance.

AMI Details



Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0b1e2eeb33ce3d66f

Free tier
eligible

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance.

Root Device Type: ebs Virtualization type: hvm

Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)
t2.micro	Variable	1	1	EBS only

Security Groups

Security group name launch-wizard-1
Description launch-wizard-1 created 2020-07-09T15:48:36.426-04:00

Type ⓘ

Protocol ⓘ

Port Range ⓘ

This security group has

Instance Details

Number of instances 1

Network vpc-fe3baa86

Infrastructure as Code

ec2.sh

```
#!/bin/bash

IP_ADDRESS="10.2.2.1"

EC2_INSTANCE=$(ec2-run-instances --instance-type
t2.micro ami-0edab43b6fa892279)

INSTANCE=$(echo ${EC2_INSTANCE} | sed 's/*INSTANCE //'
| sed 's/ .*//')

# Wait for instance to be ready
while ! ec2-describe-instances $INSTANCE | grep -q
"running"
do
    echo Waiting for $INSTANCE is to be ready...
done

# Check if instance is not provisioned and exit
if [ ! $(ec2-describe-instances $INSTANCE | grep -q
"running") ]; then
    echo Instance $INSTANCE is stopped.
    exit
fi

ec2-associate-address $IP_ADDRESS -i $INSTANCE

echo Instance $INSTANCE was created successfully!!!
```

main.tf

```
resource "aws_instance" "webserver" {
    ami          = "ami-0edab43b6fa892279"
    instance_type = "t2.micro"
}
```


Infrastructure as Code

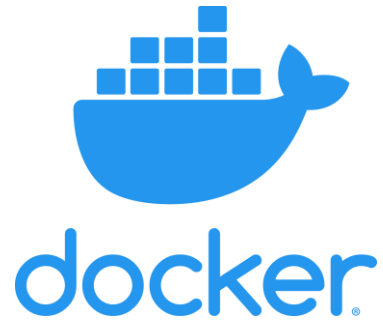
ec2.yaml

```
- amazon.aws.ec2:
  key_name: mykey
  instance_type: t2.micro
  image: ami-123456
  wait: yes
  group: webserver
  count: 3
  vpc_subnet_id: subnet-29e63245
  assign_public_ip: yes
```

main.tf

```
resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}
```

Types of IAC Tools

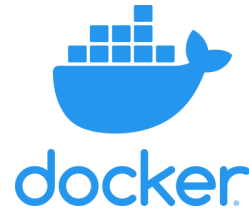


Types of IAC Tools

Configuration Management



Server Templating



Provisioning Tools



Types of IAC Tools

Configuration Management



ANSIBLE



SALTSTACK

Designed to Install and Manage Software

Maintains Standard Structure

Version Control

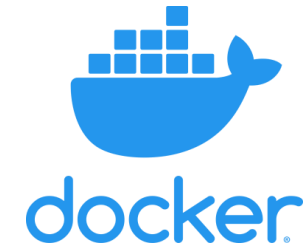
Idempotent

Server Templating Tools

Pre Installed Software and Dependencies

Virtual Machine or Docker Images

Immutable Infrastructure



Provisioning Tools

Deploy Immutable Infrastructure resources

Servers, Databases, Network Components etc.

Multiple Providers



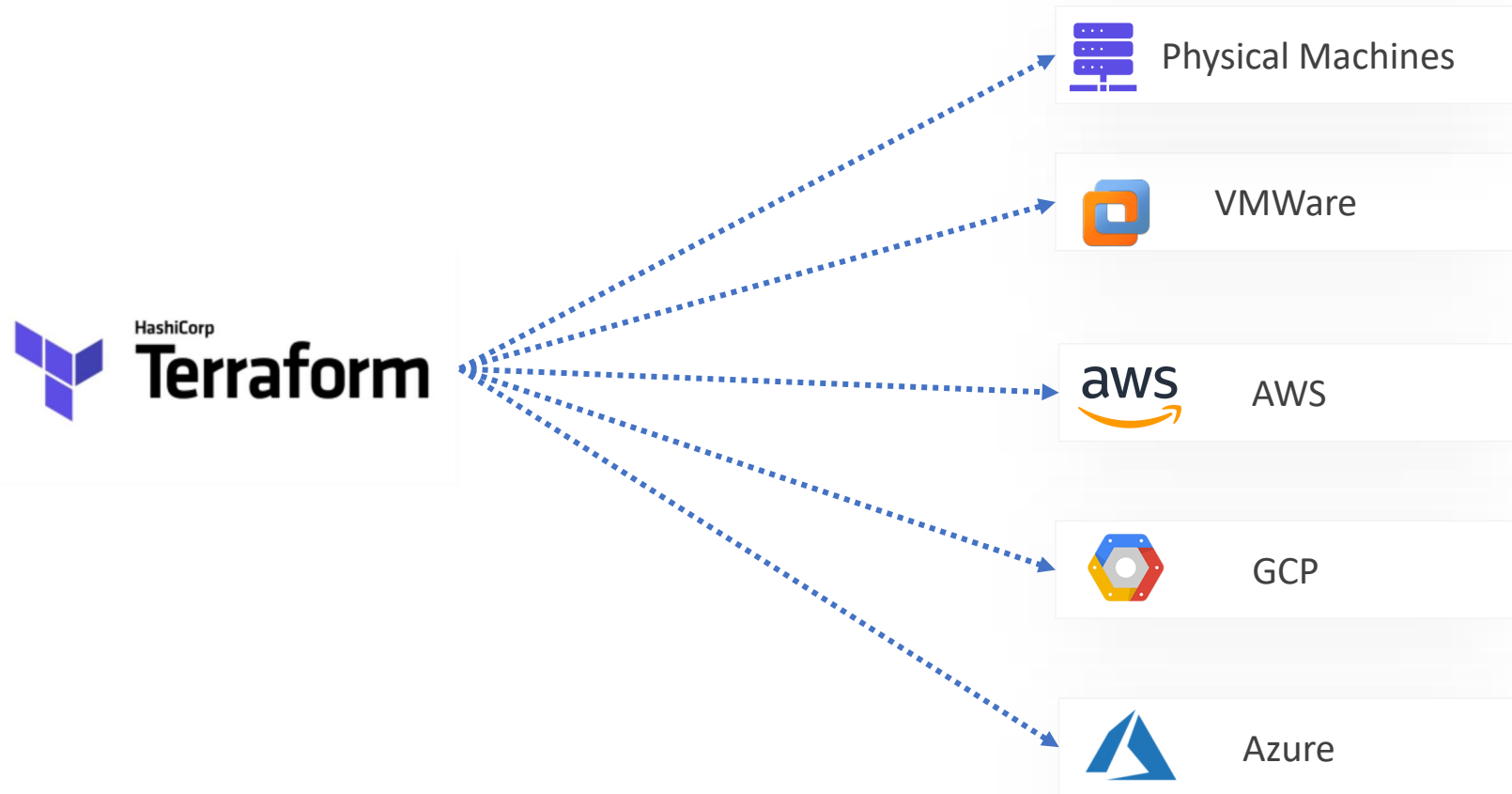


{K}ODE{K}LOUD

The background of the image is a solid blue color. In the center, there is a large, faint, light blue hexagon. Inside this hexagon, the word "Terraform" is written in a white, sans-serif font. The hexagon has a subtle gradient and is surrounded by several more faint, concentric hexagons of the same color, creating a layered effect.

Terraform

Why Terraform?



Providers

orm



Physical Machines



VMWare



AWS



GCP



Azure

BigIP

DataDog

InfluxDB

CloudFlare

Grafana

MongoDB

DNS

Auth0

MySQL

Palo Alto

Wavefront

PostgreSQL

Infoblox

Sumo Logic

VCS

HashiCorp Configuration Language

```
main.tf

resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}

resource "aws_s3_bucket" "finance" {
  bucket = "finanace-21092020"
  tags = {
    Description = "Finance and Payroll"
  }
}

resource "aws_iam_user" "admin-user" {
  name = "lucy"
  tags = {
    Description = "Team Leader"
  }
}
```

Declarative

main.tf

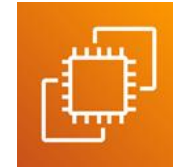
```
resource "aws_instance" "webserver" {  
  ami          = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
}  
  
resource "aws_s3_bucket" "finance" {  
  bucket = "finanace-21092020"  
  tags = {  
    Description = "Finance and Payroll"  
  }  
}  
  
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Team Leader"  
  }  
}
```

Init

Plan

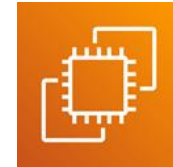
Apply

Real World Infrastructure

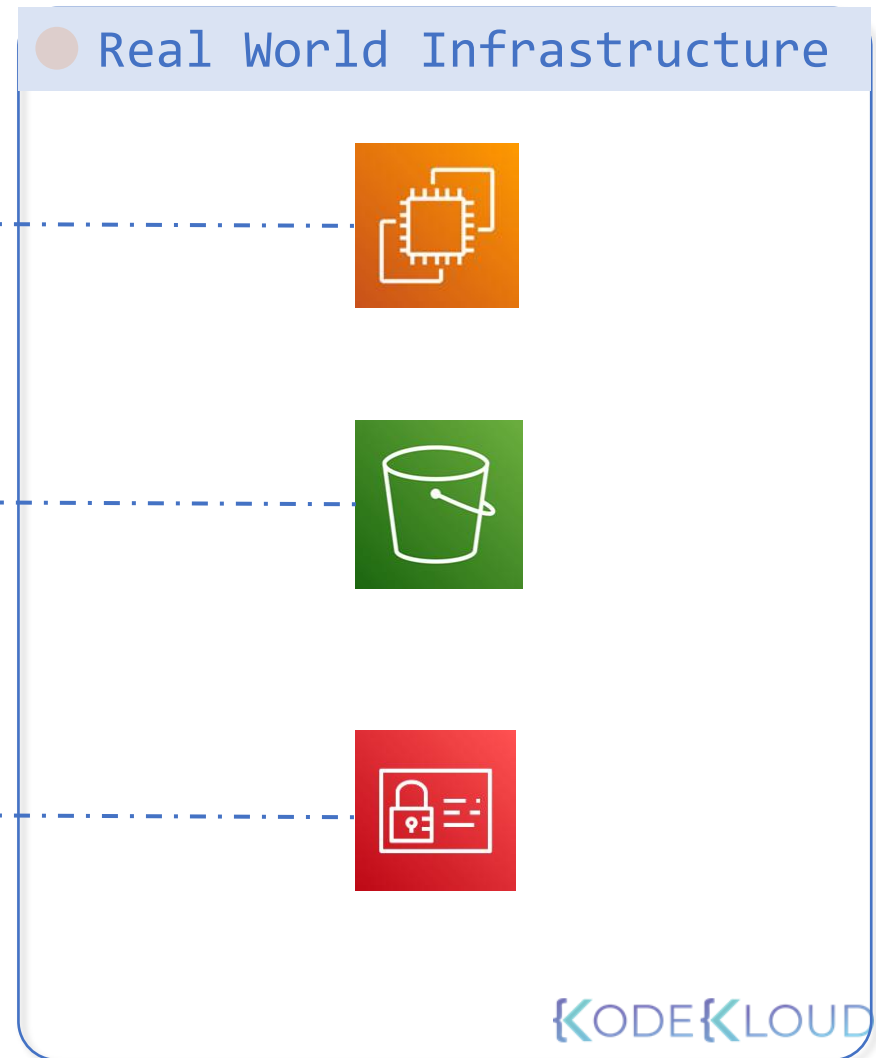
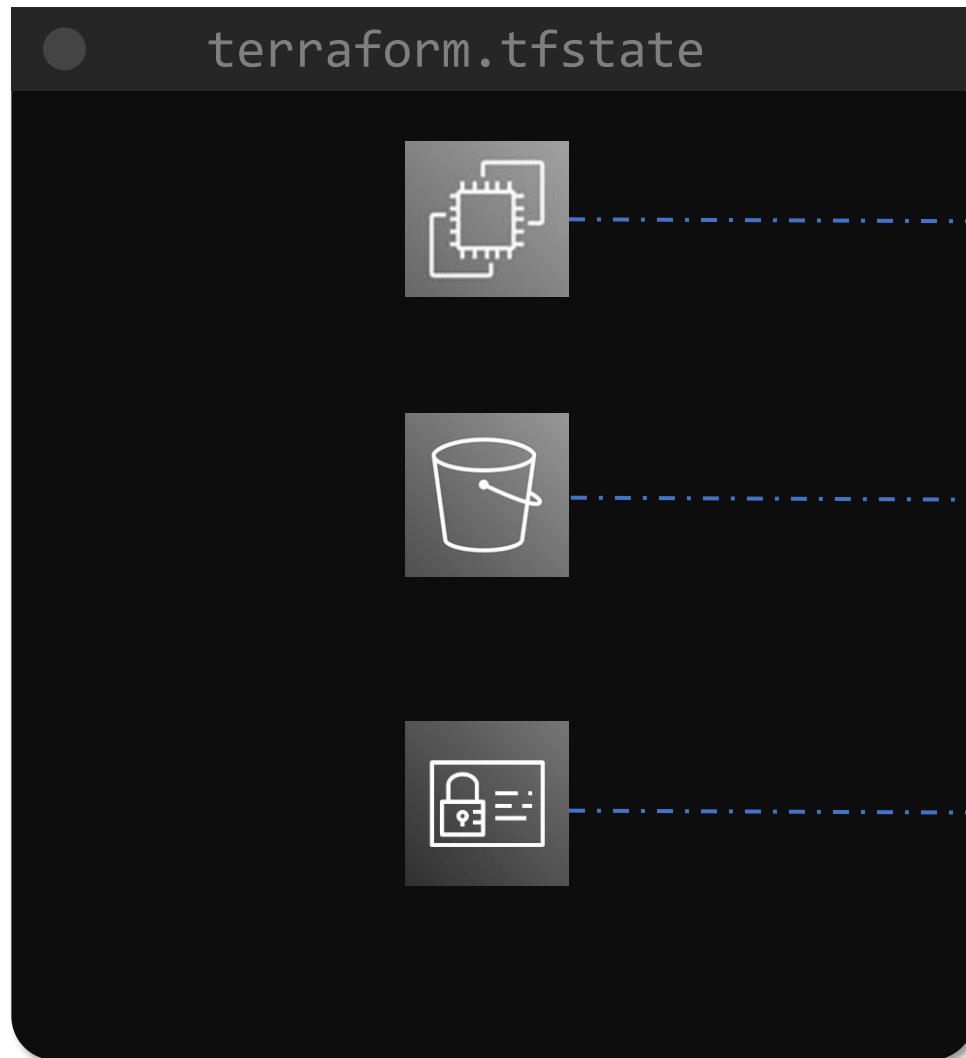


Resource

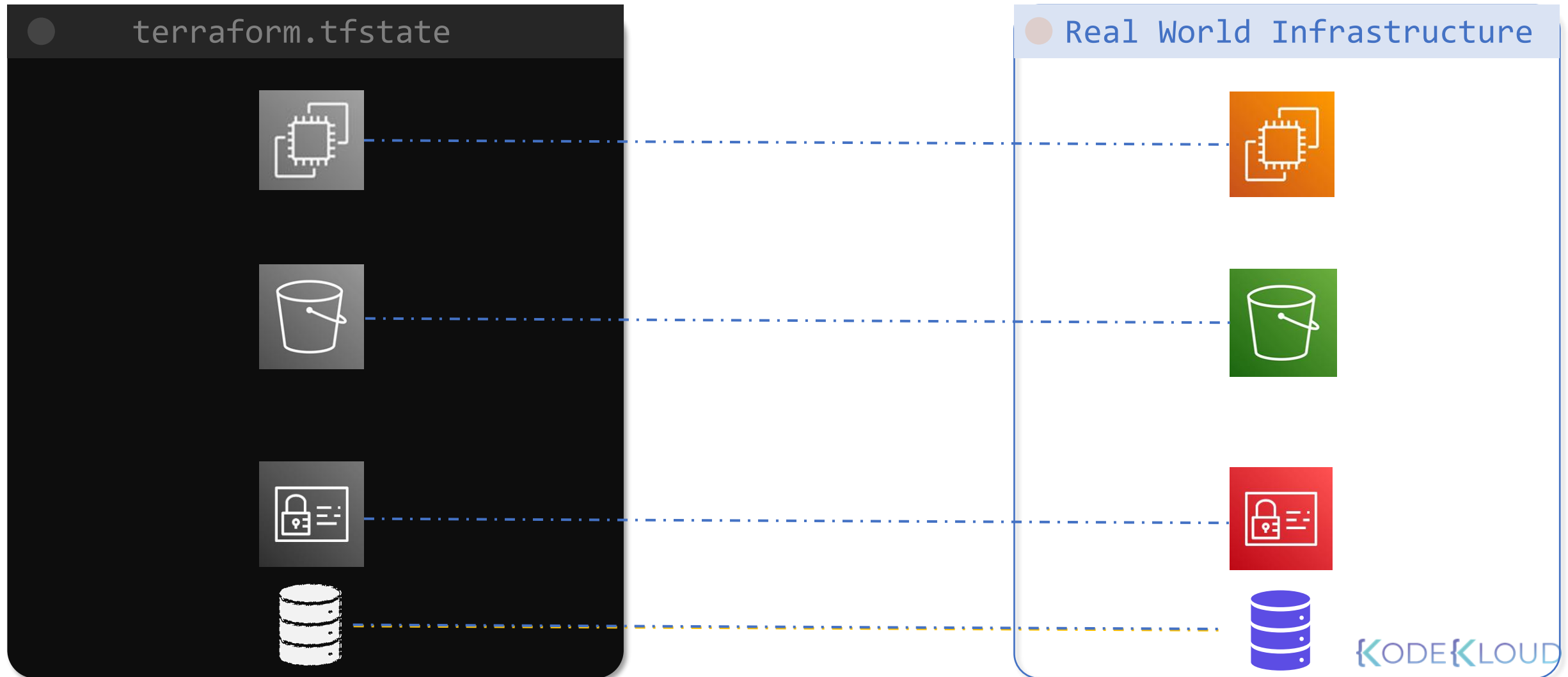
● Real World Infrastructure



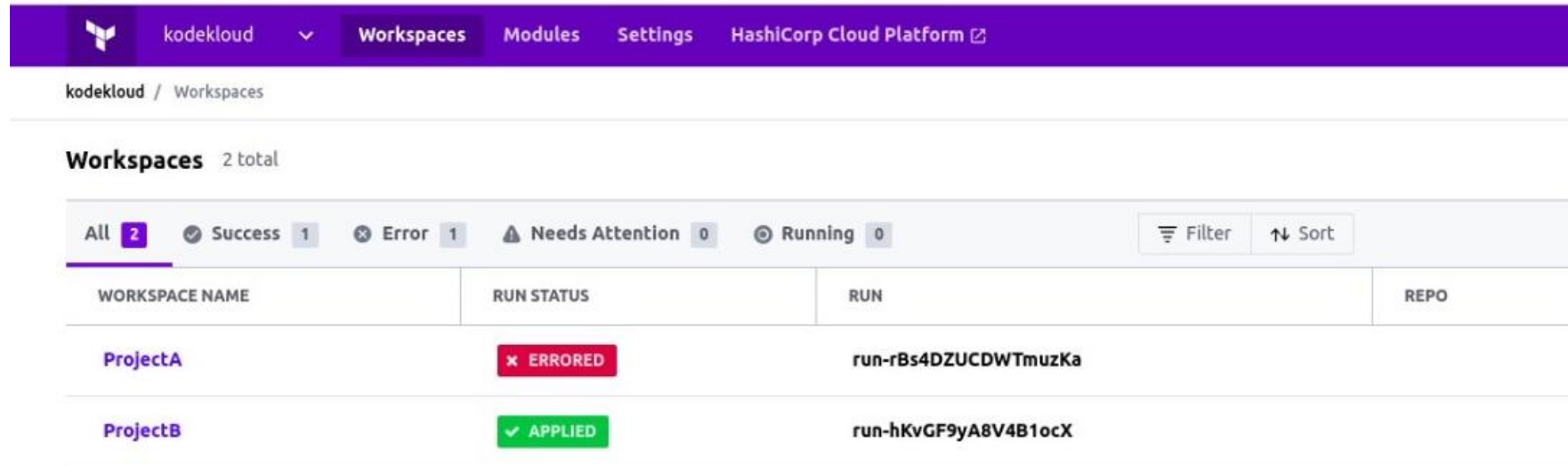
Terraform State



Terraform Import



Terraform Cloud and Terraform Enterprise



The screenshot shows the 'Workspaces' page in the KodeCloud interface. At the top, there is a navigation bar with the KodeCloud logo, a dropdown menu, and links for 'Workspaces', 'Modules', 'Settings', and 'HashiCorp Cloud Platform'. Below the navigation bar, the breadcrumb 'kodecloud / Workspaces' is visible. The main heading is 'Workspaces' followed by '2 total'. Below this, there is a filter bar with tabs for 'All' (2), 'Success' (1), 'Error' (1), 'Needs Attention' (0), and 'Running' (0). To the right of the filter bar are 'Filter' and 'Sort' buttons. The main content is a table with four columns: 'WORKSPACE NAME', 'RUN STATUS', 'RUN', and 'REPO'. The table contains two rows: 'ProjectA' with a red 'x ERRORED' status and run ID 'run-rBs4DZUCDWTmuzKa', and 'ProjectB' with a green '✓ APPLIED' status and run ID 'run-hKvGF9yA8V4B1ocX'.

WORKSPACE NAME	RUN STATUS	RUN	REPO
ProjectA	x ERRORED	run-rBs4DZUCDWTmuzKa	
ProjectB	✓ APPLIED	run-hKvGF9yA8V4B1ocX	



Installing Terraform

> _

```
$ wget https://releases.hashicorp.com/terraform/0.13.0/terraform_0.13.0_linux_amd64.zip
```

```
$ unzip terraform_0.13.0_linux_amd64.zip
```

```
$ mv terraform /usr/local/bin
```

```
$ terraform version
```

```
Terraform v0.13.0
```



macOS

64-bit



FreeBSD

32-bit | 64-bit | Arm



Linux

32-bit | 64-bit | Arm



OpenBSD

32-bit | 64-bit



Solaris

64-bit



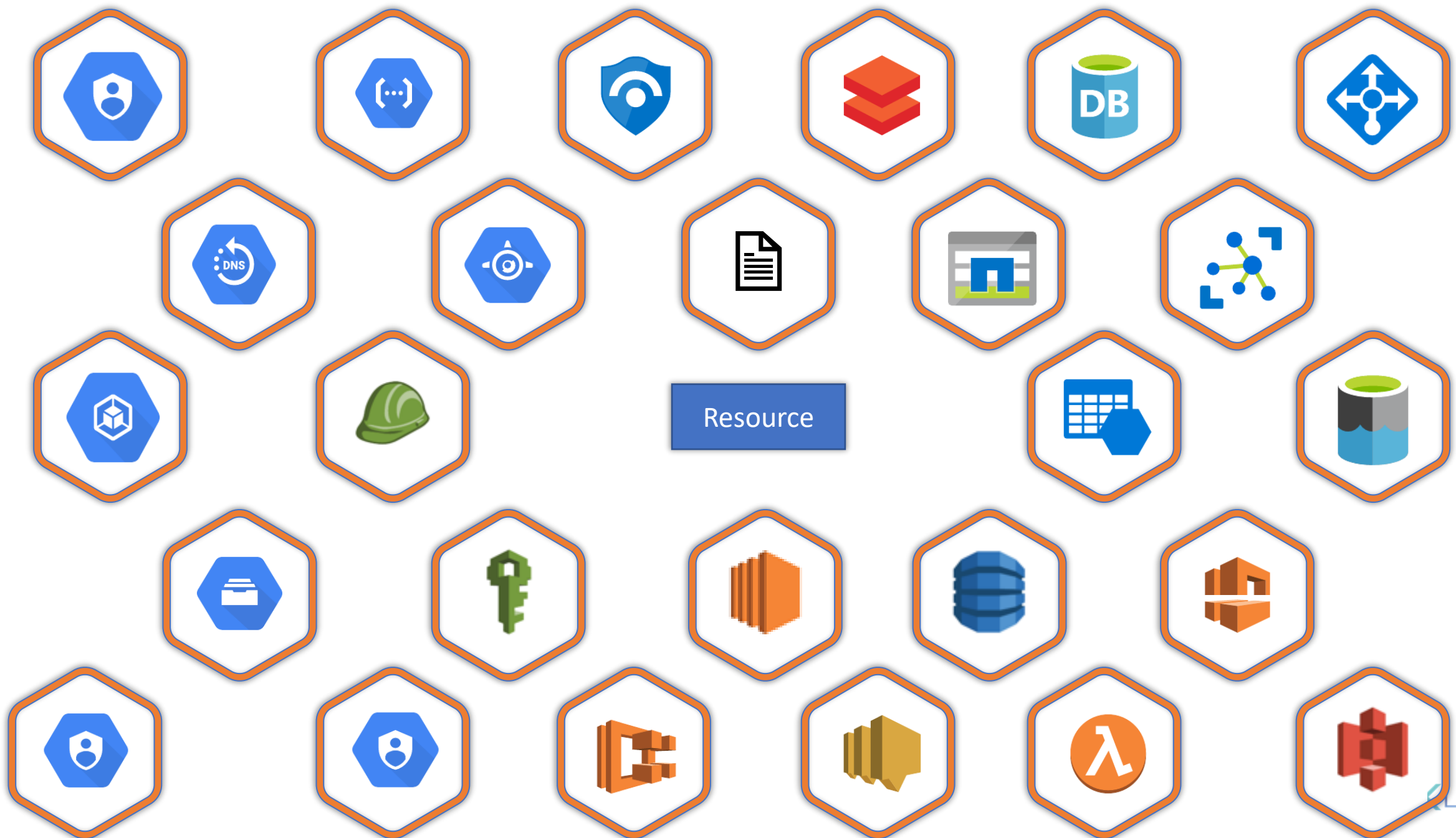
Windows

32-bit | 64-bit

HCL – Declarative Language

```
aws.tf

resource "aws_instance" "webserver" {
  ami = "ami-0c2f25c1f66a1ff4d"
  instance_type = "t2.micro"
}
```





Resource





The image features a solid blue background with a series of concentric, rounded hexagonal shapes in varying shades of blue, creating a tunnel-like effect towards the center. The text "HCL Basics" is centered within the innermost hexagon.

HCL Basics

> _

```
$ mkdir /root/terraform-local-file
```

```
$ cd /root/terraform-local-file
```

local.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}
```



local=provide
r
file_resource

Block
Name

Resource
Type

Resource
Name

FILENAME

CONTENT

Arguments

```
local.tf  
  
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}
```



```
aws-ec2.tf

resource "aws_instance" "webserver" {
  ami = "ami-0c2f25c1f66a1ff4d"
  instance_type = "t2.micro"
}
```



```
aws-s3.tf

resource "aws_s3_bucket" "data" {
    bucket = "webserver-bucket-org-2207"
    acl    = "private"
}
```



local.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}
```



```
local.tf

resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content  = "We love pets!"
}
```



```
> _
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/local: version = "~> 1.4.0"

Terraform has been successfully initialized!
```

> _

\$ terraform plan

Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be persisted to local or remote state storage.

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:

+ create

Terraform will perform the following actions:

```
# local_file.pet will be created
+ resource "local_file" "pet" {
  + content           = "We love pets!"
  + directory_permission = "0777"
  + file_permission   = "0777"
  + filename          = "/root/pets.txt"
  + id                = (known after apply)
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't specify an "-out" parameter to save this plan, so
Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.



> _

\$ terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

```
# local_file.pet will be created
+ resource "local_file" "pet" {
  + content          = "We love pets!"
  + directory_permission = "0777"
  + file_permission   = "0777"
  + filename         = "/root/pets.txt"
  + id               = (known after apply)
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

```
Enter a value: yes
local_file.new_file: Creating...
local_file.new_file: Creation complete after 0s
[id=521c5c732c78cb42cc9513ecc7c0638c4a115b55]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

\$ cat /root/pets.txt

We love pets!



> _

```
$ terraform show
```

```
# local_file.pet:
```

```
resource "local_file" "pet" {  
  content          = "We love pets!"  
  directory_permission = "0777"  
  file_permission  = "0777"  
  filename         = "/root/pets.txt"  
  id               = "cba595b7d9f94ba1107a46f3f731912d95fb3d2c"  
}
```



local=provide
r
file-resource
Resource
Type

local.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}
```



provider



resource_type

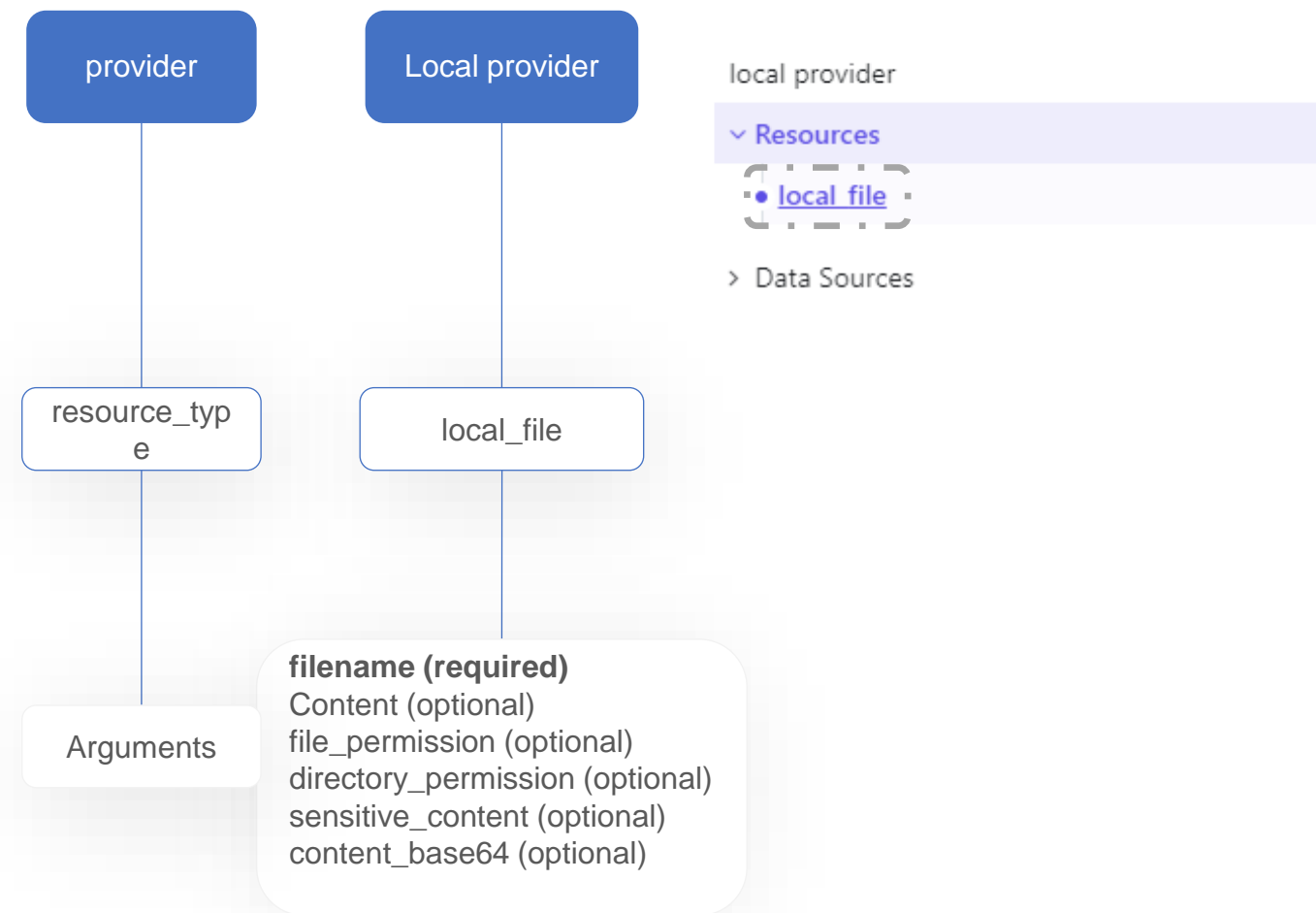


Argument-1
Argument-2
Argument-X

Argument-1
Argument-2
Argument-X

Arguments





Argument Reference

The following arguments are supported:

- `content` - (Optional) The content of file to create. Conflicts with `sensitive_content` and `content_base64`.
- `sensitive_content` - (Optional) The content of file to create. Will not be stored. Conflicts with `content` and `content_base64`.
- `content_base64` - (Optional) The base64 encoded content of the file to create. Useful when dealing with binary data. Conflicts with `content` and `sensitive_content`.
- `filename` - (Required) The path of the file to create.
- `file_permission` - (Optional) The permission to set for the created file. Expects a string. The default value is `"0777"`.
- `directory_permission` - (Optional) The permission to set for any directories created. Expects a string. The default value is `"0777"`.



The background of the slide is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in a lighter shade of blue, creating a tunnel-like effect. The text is centered within this hexagonal pattern.

Update and Destroy Infrastructure

```
local.tf

resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
  file_permission = "0700"
}
```



```
>_

$ terraform plan
```

```
local_file.pet: Refreshing state...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
```

```
-----
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement
```

```
Terraform will perform the following actions:
```

```
[# local_file.pet must be replaced]
[-/+ resource "local_file" "pet" {
    content                = "We love pets!"
    directory_permission   = "0777"
    [~ file_permission     = "0777" -> "0700" # forces replacement]
    filename               = "/root/pet.txt"
    ~ id                   =
    "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after apply)
}]

[Plan: 1 to add, 0 to change, 1 to destroy.]
```

```
-----
Note: You didn't specify an "-out" parameter to save this plan, so
Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

> _

```
$ ls -ltr /root/pets.txt  
-rwx----- 1 root root 30 Aug 17 23:20 pet.txt
```



> _

```
$ terraform apply
```

```
#local_file.pet must be replaced  
-/+ resource "local_file" "pet" {  
    content                = "We love pets!"  
    directory_permission = "0777"  
    ~ file_permission      = "0777" -> "0700" # forces replacement  
    filename               = "/root/pet.txt"  
    ~ id                  =  
    "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after apply)  
}
```

```
Plan: 1 to add, 0 to change, 1 to destroy.
```

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```
local_file.pet: Destroying...  
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]  
local_file.pet: Destruction complete after 0s  
local_file.pet: Creating...  
local_file.pet: Creation complete after 0s  
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
```

```
Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```




> _

\$ terraform destroy

local_file.pet: Refreshing state...

[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

- destroy

Terraform will perform the following actions:

```
[# local_file.pet will be destroyed .....]
- resource "local_file" "pet" {
  content          = "My favorite pet is a gold fish" -> null
  directory_permission = "0777" -> null
  file_permission   = "0700" -> null
  filename          = "/root/pet.txt" -> null
  id                = "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -
> null
}
```

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?

Terraform will destroy all your managed infrastructure, as shown above.

There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

local_file.pet: Destroying... [id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

local_file.pet: Destruction complete after 0s

```
[Destroy complete! Resources: 1 destroyed.]
```



{K}ODE{K}LOUD

Installing Terraform

> _

```
$ wget https://releases.hashicorp.com/terraform/0.13.0/terraform_0.13.0_linux_amd64.zip
$ unzip terraform_0.13.0_linux_amd64.zip
$ mv terraform /usr/local/bin
$ terraform version
Terraform v0.13.0
```



macOS
64-bit



FreeBSD
32-bit | 64-bit | Arm



Linux
32-bit | 64-bit | Arm



OpenBSD
32-bit | 64-bit



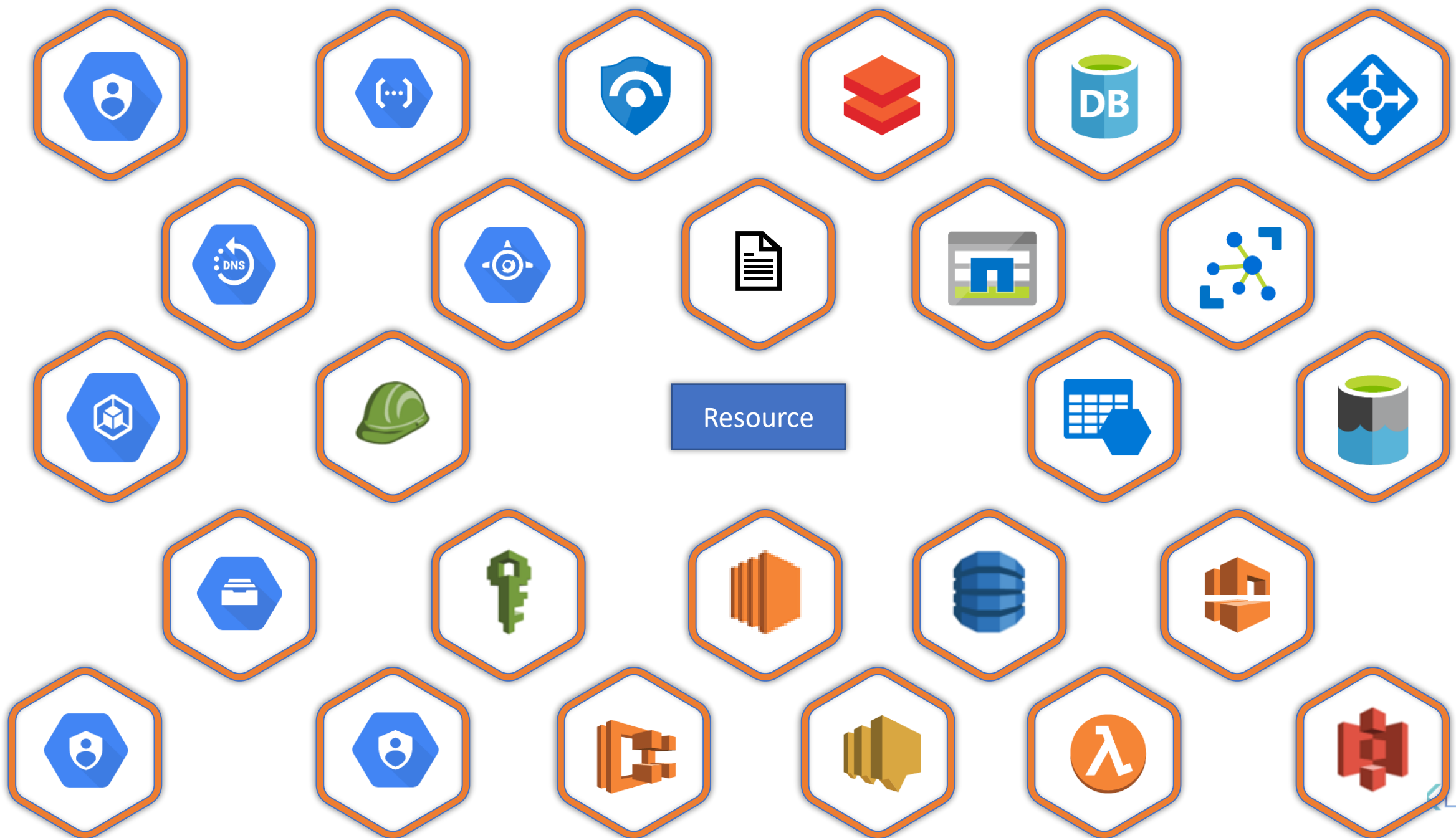
Solaris
64-bit



Windows
32-bit | 64-bit

HCL – Declarative Language

```
aws.tf  
  
resource "aws_instance" "webserver" {  
    ami = "ami-0c2f25c1f66a1ff4d"  
    instance_type = "t2.micro"  
}
```





Resource



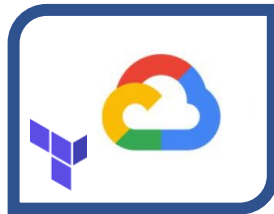


The background of the slide features a series of concentric, rounded hexagons in various shades of blue, creating a layered, tunnel-like effect. The text is centered within the innermost hexagon.

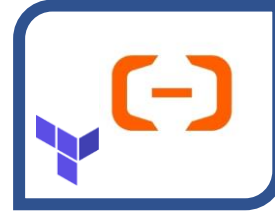
Using Terraform Providers

> _

```
$ terraform init
```



Official



Verified



bigip
by: F5Networks



heroku
by: heroku



digitalocean
by: digitalocean

Community



activedirectory



ucloud



netapp-gcp

registry.terraform.io

>_

```
$ terraform init
```

```
Initializing the backend...
```

```
Initializing provider plugins...
```

```
- Finding latest version of hashicorp/local...
```

```
- Installing hashicorp/local v2.0.0...
```

```
- Installed hashicorp/local v2.0.0 (signed by HashiCorp)
```

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking changes, we recommend adding version constraints in a `required_providers` block in your configuration, with the constraint strings suggested below.

```
* hashicorp/local: version = "~> 2.0.0"
```

```
Terraform has been successfully initialized!
```

>_

```
$ ls /root/terraform-local-file/.terraform  
plugins
```

To prevent automatic upgrades to new major versions that contain breaking changes, we recommend adding a `required_providers` block in your configuration, with the constraints below.

```
* hashicorp/local: version = "~> 2.0.0"
```

Organizational
Namespace

Type

Terraform has been successfully initialized

To prevent automatic upgrades to new major versions that may contain breaking changes, we recommend adding a `required_providers` block in your configuration, with the constraints below.

*

`registry.terraform.io/` `hashicorp/` `local`

Hostname

Organizational
Namespace

Type

Terraform has been successfully initialized

initializing provider plugins...

- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.0.0...
- Installed hashicorp/local v2.0.0 (signed by H

The following providers do not have any version configuration, so the latest version was installed.

To prevent automatic upgrades to new major versions contain breaking changes, we recommend adding version constraint required_providers block



The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in varying shades of blue, creating a tunnel-like effect. In the top-left and bottom-right corners, there are faint, light-blue geometric shapes that resemble stylized letters or symbols.

Configuration Directory

```
>_
```

```
[terraform-local-file]$ ls /root/terraform-local-file  
local.tf
```

local.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}
```

cat.tf

```
resource "local_file" "cat" {  
  filename = "/root/cat.txt"  
  content = "My favorite pet is Mr. Whiskers"  
}
```

local.tf

cat.tf

main.tf

```
resource "local_file" "pet" {  
    filename = "/root/pets.txt"  
    content = "We love pets!"  
}  
  
resource "local_file" "cat" {  
    filename = "/root/cat.txt"  
    content = "My favorite pet is Mr. Whiskers"  
}
```

File Name	Purpose
main.tf	Main configuration file containing resource definition
variables.tf	Contains variable declarations
outputs.tf	Contains outputs from resources
provider.tf	Contains Provider definition



Multiple Providers

```
main.tf

resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```



main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}  
  
resource "random_pet" "my-pet" {  
  prefix = "Mrs"  
  separator = "."  
  length = "1"  
}
```



random provider

▼ Resources

random_id

random_integer

random_password

random_pet

random_shuffle

random_string

random_uuid



Argument Reference

The following arguments are supported:

- `keepers` - (Optional) Arbitrary map of values that, w to be generated. See [the main provider documentati](#)
- `length` - (Optional) The length (in words) of the pet
- `prefix` - (Optional) A string to prefix the name with
- `separator` - (Optional) The character to separate wo

> _

```
$ terraform plan
```

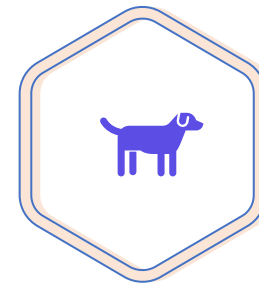
```
Refreshing Terraform state in-memory prior to plan...  
The refreshed state will be used to calculate this plan, but  
will not be  
persisted to local or remote state storage.
```

```
local_file.pet: Refreshing state...  
[id=d1a31467f206d6ea8ab1cad382bc106bf46df69e]
```

```
.  
.
```

```
# random_pet.my-pet will be created  
+ resource "random_pet" "my-pet" {  
  + id          = (known after apply)  
  + length      = 1  
  + prefix      = "Mrs"  
  + separator    = "."  
}
```

```
Plan: 1 to add, 0 to change, 0 to destroy.
```



> _

```
$ terraform init
```

```
Initializing the backend...
```

```
Initializing provider plugins...
```

```
- Using previously-installed hashicorp/local v2.0.0
```

```
- Finding latest version of hashicorp/random...
```

```
- Installing hashicorp/random v2.3.0...
```

```
- Installed hashicorp/random v2.3.0 (signed by HashiCorp)
```

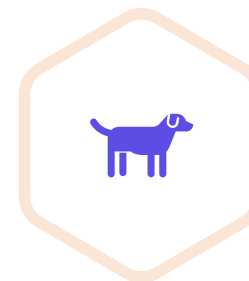
```
The following providers do not have any version constraints in  
configuration,  
so the latest version was installed.
```

```
To prevent automatic upgrades to new major versions that may contain  
breaking  
changes, we recommend adding version constraints in a required_providers  
block  
in your configuration, with the constraint strings suggested below.
```

```
* hashicorp/local: version = "~> 2.0.0"
```

```
* hashicorp/random: version = "~> 2.3.0"
```

```
Terraform has been successfully initialized!
```



> _

```
$ terraform apply
```

```
local_file.new_file: Refreshing state...  
[id=d1a31467f206d6ea8ab1cad382bc106bf46df69e]
```

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

```
# random_pet.my-pet will be created  
+ resource "random_pet" "my-pet" {  
  + id          = (known after apply)  
  + length      = 1  
  + prefix      = "Mrs"  
  + separator   = "."  
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

```
random_pet.my-pet: Creating...  
random_pet.my-pet: Creation complete after 0s [id=Mrs.hen]
```

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.



Mrs.hen



The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in varying shades of blue, creating a tunnel-like effect. In the top-left and bottom-right corners, there are faint, light-blue geometric shapes that resemble stylized letters or symbols.

Define Input
Variables

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}  
  
resource "random_pet" "my-pet" {  
  prefix = "Mrs"  
  separator = "."  
  length = "1"  
}
```

Argument	Value
filename	"/root/pets.txt"
content	"We love pets!"
prefix	"Mrs"
separator	"."
length	"1"

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}  
  
resource "random_pet" "my-pet" {  
  prefix = "Mrs"  
  separator = "."  
  length = "1"  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "We love pets!"  
}  
variable "prefix" {  
  default = "Mrs"  
}  
variable "separator" {  
  default = "."  
}  
variable "length" {  
  default = "1"  
}
```

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  content = var.content  
}  
  
resource "random_pet" "my-pet" {  
  prefix = var.prefix  
  separator = var.separator  
  length = var.length  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "We love pets!"  
}  
variable "prefix" {  
  default = "Mrs"  
}  
variable "separator" {  
  default = "."  
}  
variable "length" {  
  default = "1"  
}
```

> _

```
$ terraform apply
```

```
# local_file.pet will be created
+ resource "local_file" "pet" {
  + content          = "We love pets!"
  + directory_permission = "0777"
  + file_permission   = "0777"
  + filename         = "/root/pet.txt"
  + id               = (known after apply)
}

# random_pet.my-pet will be created
+ resource "random_pet" "my-pet" {
  + id          = (known after apply)
  + length      = 1
  + prefix      = "Mrs"
  + separator   = "."
}
```

```
Plan: 2 to add, 0 to change, 0 to destroy.
```

```
.
.
random_pet.my-pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mrs.ram]
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=f392b4bcf5db76684f719bf72061627a9a177de1]
```



main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content = var.content
}

resource "random_pet" "my-pet" {
  prefix = var.prefix
  separator = var.separator
  length = var.length
}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}

variable "content" {
  default = "My favorite pet is Mrs. Whiskers"
}

variable "prefix" {
  default = "Mrs"
}

variable "separator" {
  default = "."
}

variable "length" {
  default = "2"
}
```

>_

\$ terraform apply

Terraform will perform the following actions:

```
-/+ resource "local_file" "pet" {  
  ~ content          = "We love pets!" -> "My favorite pet is Mrs. Whiskers!" #  
  forces replacement  
    directory_permission = "0777"  
    file_permission      = "0777"  
    filename             = "/root/pet.txt"  
  ~ id               = "bc9cabef1d8b0071d3c4ae9959a9c328f35fe697" -> (known after  
apply)  
}
```

```
# random_pet.my-pet must be replaced  
-/+ resource "random_pet" "my-pet" {  
  ~ id          = "Mrs.Hen" -> (known after apply)  
  ~ length      = 1 -> 2 # forces replacement  
  prefix       = "Mrs"  
  separator    = "."  
}
```

Plan: 2 to add, 0 to change, 2 to destroy.

random_pet.my-pet: Destroying... [id=Mrs.hen]

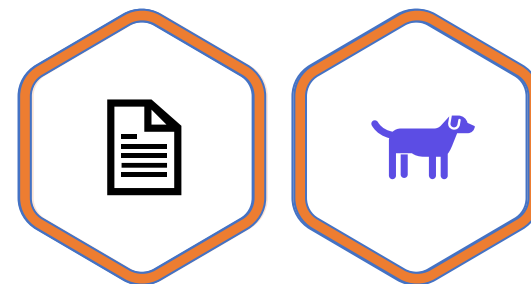
random_pet.my-pet: Destruction complete after 0s

local_file.pet: Destroying... [id=bc9cabef1d8b0071d3c4ae9959a9c328f35fe697]

local_file.pet: Destruction complete after 0s

random_pet.my-pet: Creating...

local_file.pet: Creating...



main.tf

```
resource "aws_instance" "webserver" {  
    ami            = var.ami  
    instance_type = var.instance_type  
}
```

variables.tf

```
variable "ami" {  
    default = "ami-0edab43b6fa892279"  
}  
variable "instance_type" {  
    default = "t2.micro"  
}
```



{K}ODE{K}LOUD



Understanding the Variable Block

variables.tf

```
variable "filename" {  
    default = "/root/pets.txt"  
}  
variable "content" {  
    default = "I love pets!"  
}  
variable "prefix" {  
    default = "Mrs"  
}  
variable "separator" {  
    default = "."  
}  
variable "length" {  
    default = "1"  
}
```

variables.tf

```
variable "filename" {  
    default = "/root/pets.txt"  
    type = string  
    description = "the path of local file"  
  
}  
variable "content" {  
    default = "I love pets!"  
    type = string  
    description = "the content of the file"  
  
}  
variable "prefix" {  
    default = "Mrs"  
    type = string  
    description = "the prefix to be set"  
  
}  
variable "separator" {  
    default = "."
```

variables.tf

```
variable "filename" {  
    default = "/root/pets.txt"  
    type = string  
    description = "the path of local file"  
}  
  
variable "content" {  
    default = "I love pets!"  
    type = string  
    description = "the content of the file"  
}  
  
variable "prefix" {  
    default = "Mrs"  
    type = string  
    description = "the prefix to be set"  
}  
  
variable "separator" {  
    default = "."  
}
```

Type	Example
string	"/root/pets.txt"
number	1
bool	true/false
any	Default Value

variables.tf

```
variable "length" {  
    default = "2"  
    type = number  
    description = "length of the pet name"  
}  
  
variable "password_change" {  
    default = "true"  
    type = bool  
}
```


Type	Example
string	"/root/pets.txt"
number	1
bool	true/false
any	Default Value
list	["cat", "dog"]
map	pet1 = cat pet2 = dog
object	Complex Data Structure
tuple	Complex Data Structure

List

variables.tf

```
variable "prefix" {  
  default = ["Mr", "Mrs", "Sir"]  
  type = list 0 1 2  
}
```

main.tf

```
resource "random_pet" "my-pet" {  
  prefix = var.prefix[0]  
}
```

Index	Value
0	Mr
1	Mrs
2	Sir

Map

variables.tf

```
variable file-content {  
  type      = map  
  default   = {  
    "statement1" = "We love pets!"  
    "statement2" = "We love animals!"  
  }  
}
```

maint.tf

```
resource local_file my-pet {  
  filename = "/root/pets.txt"  
  content  = var.file-content["statement2"]  
}
```

Key	Value
statement1	We love pets!
statement2	We love animals!

List of a Type

variables.tf

```
variable "prefix" {  
  default = ["Mr", "Mrs", "Sir"]  
  type = list(string)  
}
```

variables.tf

```
variable "prefix" {  
  default = ["Mr", "Mrs", "Sir"]  
  type = list(number)  
}
```

variables.tf

```
variable "prefix" {  
  default = ["1", "2", "3"]  
  type = list(number)  
}
```

>_

```
$ terraform plan
```

```
Error: Invalid default value for variable
```

```
on variables.tf line 3, in variable "prefix":  
  3:   default      = ["Mr", "Mrs", "Sir"]
```

```
This default value is not compatible with the  
variable's type constraint: a number is required.
```

Map of a Type

variables.tf

```
variable "cats" {  
  default = {  
    "color" = "brown"  
    "name"  = "bella"  
  }  
  type = map(string)  
}
```

variables.tf

```
variable "pet_count" {  
  default = {  
    "dogs" = "3"  
    "cats" = "1"  
    "goldfish" = "2"  
  }  
  type = map(number)  
}
```

Set

variables.tf

```
variable "prefix" {  
  default = ["Mr", "Mrs", "Sir"]  
  type = set(string)  
}
```

variables.tf

```
variable "prefix" {  
  default = ["Mr", "Mrs", "Sir", "Sir"]  
  type = set(string)  
}
```

variables.tf

```
variable "fruit" {  
  default = ["apple", "banana"]  
  type = set(string)  
}
```

variables.tf

```
variable "fruit" {  
  default = ["apple", "banana", "banana"]  
  type = set(string)  
}
```

variables.tf

```
variable "age" {  
  default = ["10", "12", "15"]  
  type = set(number)  
}
```

variables.tf

```
variable "age" {  
  default = ["10", "12", "15", "10"]  
  type = set(number)  
}
```

Objects

Key	Example	Type
name	bella	string
color	brown	string
age	7	number
food	["fish", "chicken", "turkey"]	list
favorite_pet	true	bool

```
variables.tf

variable "bella" {
  type = object({
    name = string
    color = string
    age = number
    food = list(string)
    favorite_pet = bool
  })

  default = {
    name = "bella"
    color = "brown"
    age = 7
    food = ["fish", "chicken", "turkey"]
    favorite_pet = true
  }
}
```

Tuples

variables.tf

```
variable kitty {  
  type      = tuple([string, number, bool])  
  default   = ["cat", 7, true]  
}
```

variables.tf

```
variable kitty {  
  type      = tuple([string, number, bool])  
  default   = ["cat", 7, true, "dog"]  
}
```

>_

\$ terraform plan

Error: Invalid default value for variable

on variables.tf line 3, in variable "kitty":
3: default = ["cat", 7, true, "dog"]

This default value is not compatible with the
variable's type constraint:
tuple required.



{K}ODE{K}LOUD

The background of the slide features a solid blue color with several concentric, semi-transparent hexagonal shapes centered in the middle. These hexagons vary in opacity, creating a layered effect. There are also some angular, geometric shapes in the corners, such as a light blue shape in the top-left and bottom-right, and a darker blue shape in the bottom-right corner.

Using Variables in Terraform

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  content = var.content  
}  
  
resource "random_pet" "my-pet" {  
  prefix = var.prefix  
  separator = var.separator  
  length = var.length  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "We love pets!"  
}  
variable "prefix" {  
  default = "Mrs"  
}  
variable "separator" {  
  default = "."  
}  
variable "length" {  
  default = 2  
}
```

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content = var.content
}

resource "random_pet" "my-pet" {
  prefix = var.prefix
  separator = var.separator
  length = var.length
}
```

variables.tf

```
variable "filename" {

}

variable "content" {

}

variable "prefix" {

}

variable "separator" {

}

variable "length" {

}
```

Interactive Mode

> _

```
$ terraform apply
```

```
var.content
```

```
Enter a value: We love Pets!
```

```
var.filename
```

```
Enter a value: /root/pets.txt
```

```
var.length
```

```
Enter a value: 2
```

```
var.prefix
```

```
Enter a value: Mrs.
```

```
var.separator
```

```
Enter a value: .
```

Command Line Flags

> _

```
$ terraform apply -var "filename=/root/pets.txt" -var "content=We love  
Pets!" -var "prefix=Mrs" -var "separator=." -var "length=2"
```

Environment Variables

> _

```
$ export TF_VAR_filename="/root/pets.txt"  
$ export TF_VAR_content="We love pets!"  
$ export TF_VAR_prefix="Mrs"  
$ export TF_VAR_separator="."  
$ export TF_VAR_length="2"  
$ terraform apply
```

Variable Definition Files

```
terraform.tfvars  
  
filename = "/root/pets.txt"  
content = "We love pets!"  
prefix = "Mrs"  
separator = "."  
length = "2"
```

```
> _
```

```
$ terraform apply -var-file variables.tfvars
```

Automatically Loaded

terraform.tfvars

|

terraform.tfvars.json

*.auto.tfvars

|

*.auto.tfvars.json

Variable Definition Precedence

main.tf

```
resource local_file pet {  
  filename = var.filename  
}
```

variables.tf

```
variable filename {  
  type = string  
}
```

>_

```
$ export TF_VAR_filename="/root/cats.txt" ?
```

terraform.tfvars

```
filename = "/root/pets.txt" ?
```

variable.auto.tfvars

```
filename = "/root/mypet.txt" ?
```

>_

```
$ terraform apply -var "filename=/root/best-pet.txt" ?
```

Variable Definition Precedence

Order	Option
1	Environment Variables
2	terraform.tfvars
3	*.auto.tfvars (alphabetical order)
4	-var or -var-file (command-line flags)



>_

```
$ export TF_VAR_filename="/root/cats.txt" 1
```



terraform.tfvars

```
filename = "/root/pets.txt" 2
```



variable.auto.tfvars

```
filename = "/root/mypet.txt" 3
```

>_

```
$ terraform apply -var "filename=/root/best-pet.txt" 4
```



The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in a lighter shade of blue. In the top-left and bottom-right corners, there are geometric shapes resembling folded paper or triangles in a slightly different shade of blue.

Resource Attribute Reference

```
main.tf

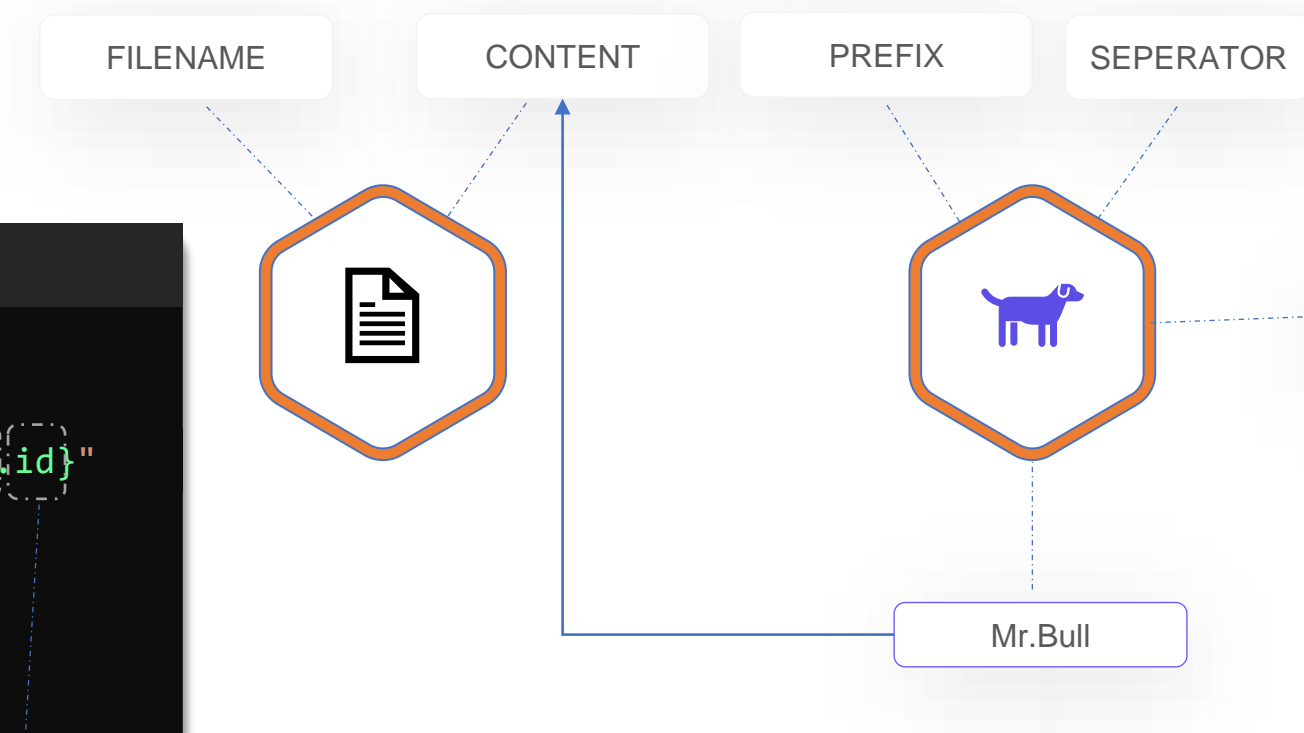
resource "local_file" "pet" {
  filename = var.filename
  content = "My favorite pet is ${random_pet.my-pet.id}"
}

resource "random_pet" "my-pet" {
  prefix = var.prefix
  separator = var.separator
  length = var.length
}
```

```
>_

random_pet.my-pet: Creating...
local_file.pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mr.bull]
local_file.pet: Creation complete after 0s
[id=059090e865809f9b6debfd7aebf48fdce2220a6]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```



Attribute Reference

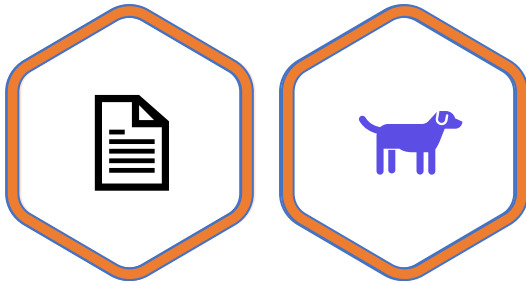
The following attributes are supported:

- `id` - (string) The random pet name

```
_file" "pet" {  
var.filename
```

My favorite pet is **Mr.Bull**"

```
m_pet" "my-pet" {  
.prefix  
var.separator  
.length
```



>_

```
$ terraform apply
```

```
.
.
.
  # local_file.pet must be replaced
-/+ resource "local_file" "pet" {
  ~ content          = "My favorite pet is Mrs.Cat!" ->
  "My favorite pet is Mr.bull" # forces replacement
  ~ directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/roots/pets.txt"
  ~ id                 =
  "98af5244e23508cffd4a0c3c46546821c4ccbdb0" -> (known after
  apply)
}
.
.
local_file.pet: Destroying...
[id=98af5244e23508cffd4a0c3c46546821c4ccbdb0]
local_file.pet: Destruction complete after 0s
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=e56101d304de7cf1b1001102923c6bdeaa60c523]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```



The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in varying shades of blue, creating a tunnel-like effect. In the top-left and bottom-right corners, there are faint, light-blue geometric shapes that resemble stylized letters or abstract forms.

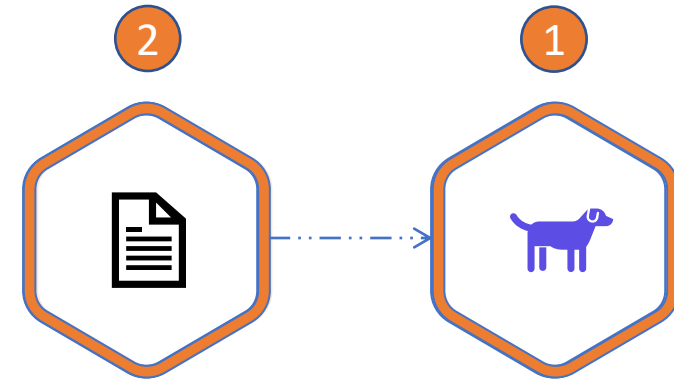
Resource Dependencies

Implicit Dependency

```
main.tf

resource "local_file" "pet" {
  filename = var.filename
  content = "My favorite pet is: ${random_pet.my-pet.id}"
}

resource "random_pet" "my-pet" {
  prefix = var.prefix
  separator = var.separator
  length = var.length
}
```



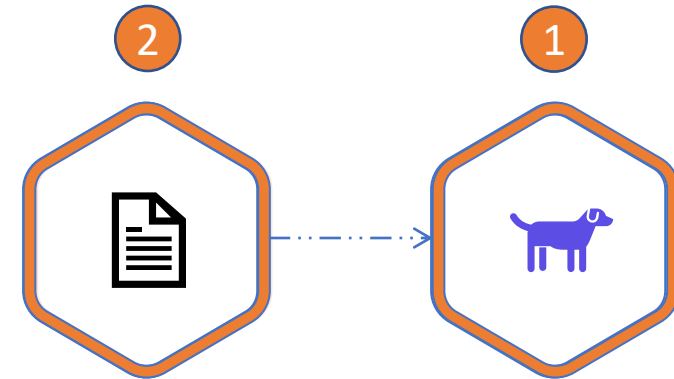
Explicit Dependency

```
main.tf

resource "local_file" "pet" {
  filename = var.filename
  content = "My favorite pet is Mr.Cat"

  depends_on = [
    random_pet.my-pet
  ]
}

resource "random_pet" "my-pet" {
  prefix = var.prefix
  separator = var.separator
  length = var.length
}
```





Output Variables

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content = "My favorite pet is ${random_pet.my-pet.id}"
}

resource "random_pet" "my-pet" {
  prefix = var.prefix
  separator = var.separator
  length = var.length
}

output pet-name {
  value      = random_pet.my-pet.id
  description = "Record the value of pet ID generated by the random_pet resource"
}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}

variable "content" {
  default = "I love pets!"
}

variable "prefix" {
  default = "Mrs"
}

variable "separator" {
  default = "."
}

variable "length" {
  default = "1"
}
```

```
output "<variable_name>" {
  value = "<variable_value>"
  <arguments>
}
```



> _

```
$ terraform apply
```

```
·  
·
```

```
Outputs:
```

```
pet-name = Mrs.gibbon
```

> _

```
$ terraform output
```

```
pet-name = Mrs.gibbon
```

> _

```
$ terraform output pet-name
```

```
Mrs.gibbon
```



Output Variable



ANSIBLE



SHELL SCRIPTS



{K}ODE{K}LOUD

The background of the slide is a solid blue color. It features several concentric, semi-transparent hexagonal shapes in the center, creating a layered effect. There are also some angular, semi-transparent blue shapes in the corners, particularly on the left and bottom right.

Introduction to Terraform State

```
> _
```

```
$ ls terraform-local-file  
main.tf variables.tf
```

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  content  = var.content  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "I love pets!"  
}
```



```
> _
```

```
$ cd terraform-local-file
```

```
[terraform-local-file]$ terraform init
```

```
Initializing the backend...
```

```
Initializing provider plugins...
```

- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration, so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking changes, we recommend adding version constraints in a `required_providers` block in your configuration, with the constraint strings suggested below.

```
* hashicorp/local: version = "~> 1.4.0"
```

```
Terraform has been successfully initialized!
```

```
>_
```

```
$ ls terraform-local-file  
main.tf variables.tf
```

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  content  = var.content  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "I love pets!"  
}
```



```
>_
```

```
[terraform-local-file]$ terraform plan
```

Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan,
persisted to local or remote state storage.

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbol:

[+ create]

Terraform will perform the following actions:

```
# local_file.pet will be created  
+ resource "local_file" "pet" {  
  + content              = "I love pets!"  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename             = "/root/pets.txt"  
  + id                   = (known after apply)  
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't specify an "-out" parameter to save this plan.

>_

```
$ ls terraform-local-file  
main.tf variables.tf
```

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  content  = var.content  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "I love pets!"  
}
```



>_

```
[terraform-local-file]$ terraform apply
```

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

```
# local_file.pet will be created  
+ resource "local_file" "pet" {  
  + content                = "I love pets!"  
  + directory_permission  = "0777"  
  + file_permission       = "0777"  
  + filename               = "/root/pets.txt"  
  + id                    = (known after apply)  
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```
local_file.pet: Creating...  
local_file.pet: Creation complete after 0s  
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]
```

> _

```
[terraform-local-file]$ cat /root/pets
```

I love pets!



> _

```
[terraform-local-file]$ terraform apply
```

local_file.pet: Refreshing state...

[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]

Apply complete! Resources: 0 added, 0 changed, 0 destroyed

> _

```
[terraform-local-file]$ ls  
main.tf variables.tf terraform.tfstate
```



> _

```
[terraform-local-file]$ cat terraform.tfstate  
{  
  "version": 4,  
  "terraform_version": "0.13.0",  
  "serial": 1,  
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31"  
  "outputs": {},  
  "resources": [  
    {  
      "mode": "managed",  
      "type": "local_file",  
      "name": "pet",  
      "provider":  
"provider[\"registry.terraform.io/hashicorp/local\"]",  
      "instances": [  
        {  
          "schema_version": 0,  
          "attributes": {  
            "content": "I love pets!",  
            "content_base64": null,  
            "directory_permission": "0777",  
            "file_permission": "0777",  
            "filename": "/root/pets.txt",  
            "id":  
"7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68",  
            "sensitive_content": null  
          },  
          "private": "bnVsbA=="  
        }  
      ]  
    }  
  ]  
}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "We love pets!"
}
```



>_

\$ terraform plan

Refreshing Terraform state in-memory prior to plan...

The refreshed state will be used to calculate this plan, but will not be persisted to local or remote state storage.

local_file.pet: Refreshing state...

[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]

.
.

[Output Truncated]

>_

```
[terraform-local-file]$ cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 1,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31"
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "provider":
        "provider[\"registry.terraform.io/hashicorp/local\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "I love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            "file_permission": "0777",
            "filename": "/root/pets.txt",
            "id":
              "7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68",
            "sensitive_content": null
          },
          "private": "bnVsbA=="
        }
      ]
    }
  ]
}
```


variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "We love pets!"
}
```



>_

\$ terraform apply

local_file.pet: Refreshing state...
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]

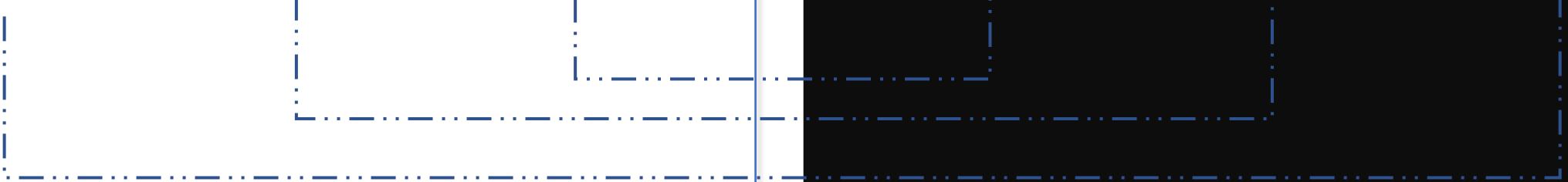
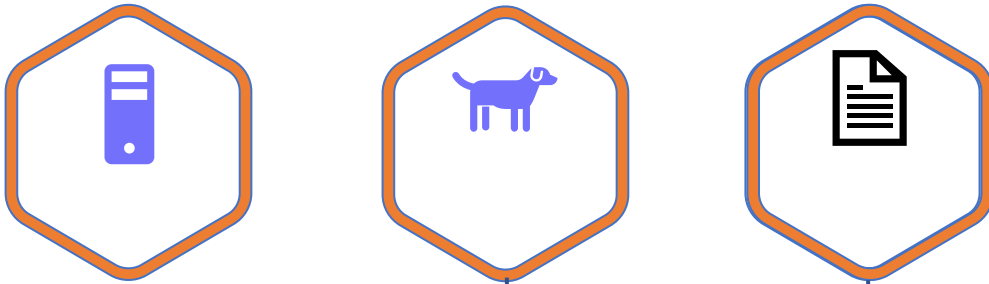
Terraform will perform the following actions:

```
# local_file.pet must be replaced
-/+ resource "local_file" "pet" {
  ~ content          = "I love pets!" -
  > "We love pets!" # forces replacement
    directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/root/pets.txt"
  ~ id               =
  "7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68" ->
  (known after apply)
}
```

>_

```
[terraform-local-file]$ cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 1,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "provider":
        "provider[\"registry.terraform.io/hashicorp/local\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            "file_permission": "0777",
            "filename": "/root/pets.txt",
            "id":
              "7e4db4fbfdbb108bdd04692602bae3e9bc4d1c14",
            "sensitive_content": null
          },
          "private": "bnVsbA=="
        }
      ]
    }
  ]
}
```

Real World Infrastructure



terraform.tfstate



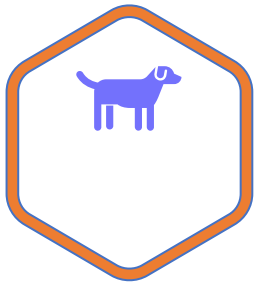


{K}ODE{K}LOUD

The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in a lighter shade of blue. In the top-left and bottom-right corners, there are geometric shapes resembling folded paper or triangles in a slightly different shade of blue.

Purpose of State

Real World Infrastructure



terraform.tfstate



id=aabbcc



id=eeddff



id=gghhhii

Tracking Metadata

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pet.txt"  
  content  = "My favorite pet is ${random_pet.my-pet.id}!"  
}  
resource "random_pet" "my-pet" {  
  length = 1  
}  
resource "local_file" "cat" {  
  filename = "/root/cat.txt"  
  content  = "I like cats too!"  
}
```



Tracking Metadata

> _

```
$ terraform apply
```

```
.  
. .  
. .
```

```
Plan: 3 to add, 0 to change, 0 to destroy.
```

```
Do you want to perform these actions?
```

```
Terraform will perform the actions described above.
```

```
Only 'yes' will be accepted to approve.
```

```
Enter a value: yes
```

```
local_file.cat: Creating...
```

```
random_pet.my-pet: Creating...
```

```
local_file.cat: Creation complete after 0s
```

```
[id=fe44888891fc40342313bc44a1f1a8986520c89]
```

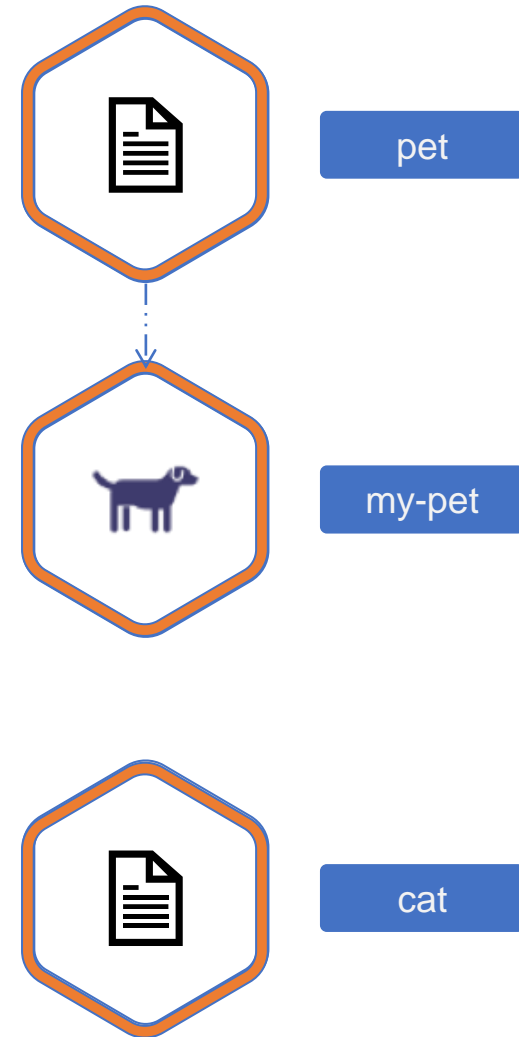
```
random_pet.my-pet: Creation complete after 0s [id=yak]
```

```
local_file.pet: Creating...
```

```
local_file.pet: Creation complete after 0s
```

```
[id=28b373c6c1fa3fce132a518eadd0175c98f37f20]
```

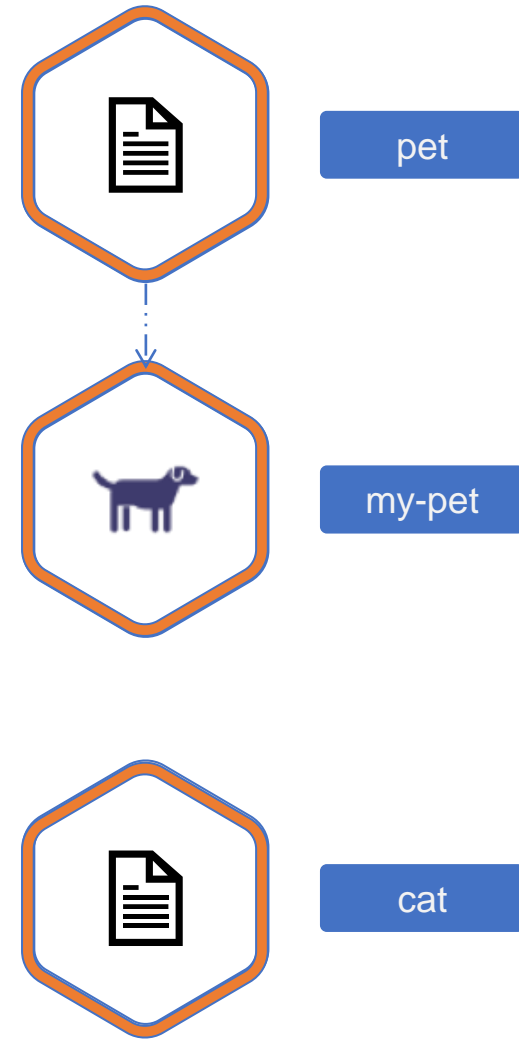
```
Apply complete! Resources: 3 added, 0 changed, 0  
destroyed.
```



Tracking Metadata

main.tf

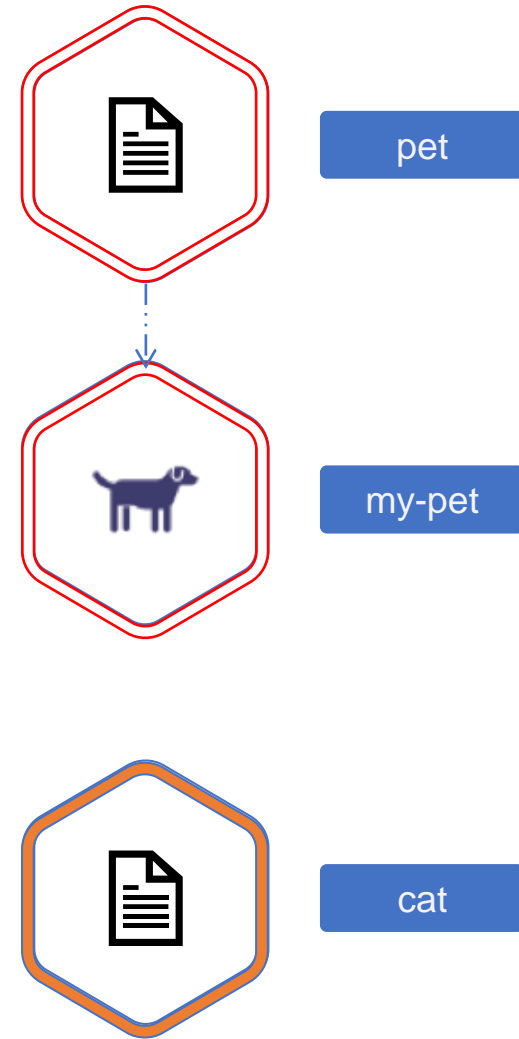
```
resource "local_file" "pet" {  
  filename = "/root/pet.txt"  
  content  = "My favorite pet is ${random_pet.my-pet.id}!"  
}  
resource "random_pet" "my-pet" {  
  length = 1  
}  
resource "local_file" "cat" {  
  filename = "/root/cat.txt"  
  content  = "I like cats too!"  
}
```



Tracking Metadata

main.tf

```
resource "local_file" "cat" {  
  filename = "/root/cat.txt"  
  content  = "I like cats too!"  
}
```



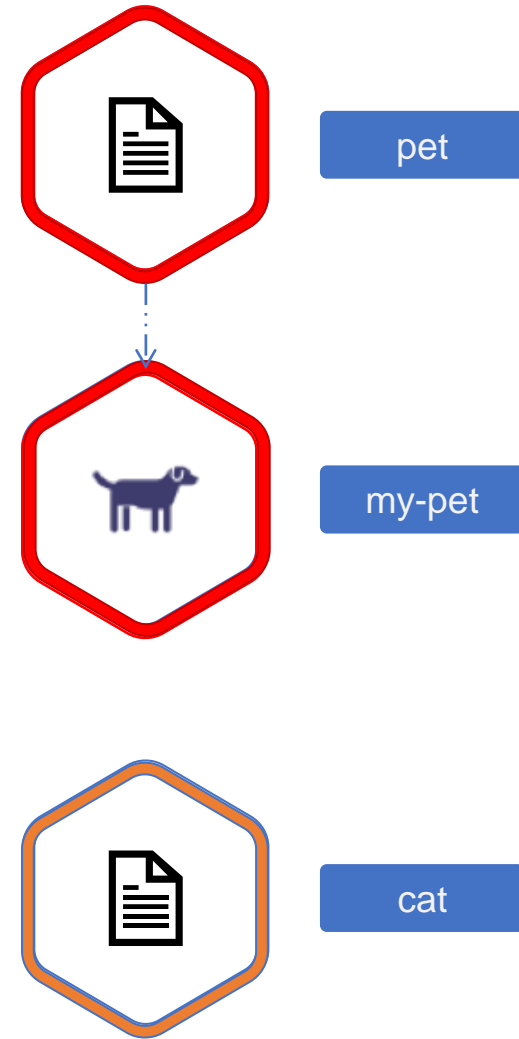
Tracking Metadata

main.tf

```
resource "local_file" "cat" {  
  filename = "/root/cat.txt"  
  content  = "I like cats too!"  
}
```

>_

```
$ cat terraform.tfstate  
{  
  "mode": "managed",  
  "type": "local_file",  
  "name": "pet",  
  "instances": [  
    {  
      "schema_version": 0,  
      "attributes": {  
        "content": "My favorite pet is yak!",  
      },  
      "private": "bnVsbA==",  
      "dependencies": [  
        "random_pet.my-pet"  
      ]  
    }  
  ]  
}
```



Tracking Metadata

main.tf

```
resource "local_file" "cat" {  
  filename = "/root/cat.txt"  
  content  = "I like cats too!"  
}
```

> _

```
$ terraform apply
```

```
Plan: 0 to add, 0 to change, 2 to destroy.
```

```
Do you want to perform these actions?
```

```
Terraform will perform the actions described above.
```

```
Only 'yes' will be accepted to approve.
```

```
Enter a value: yes
```

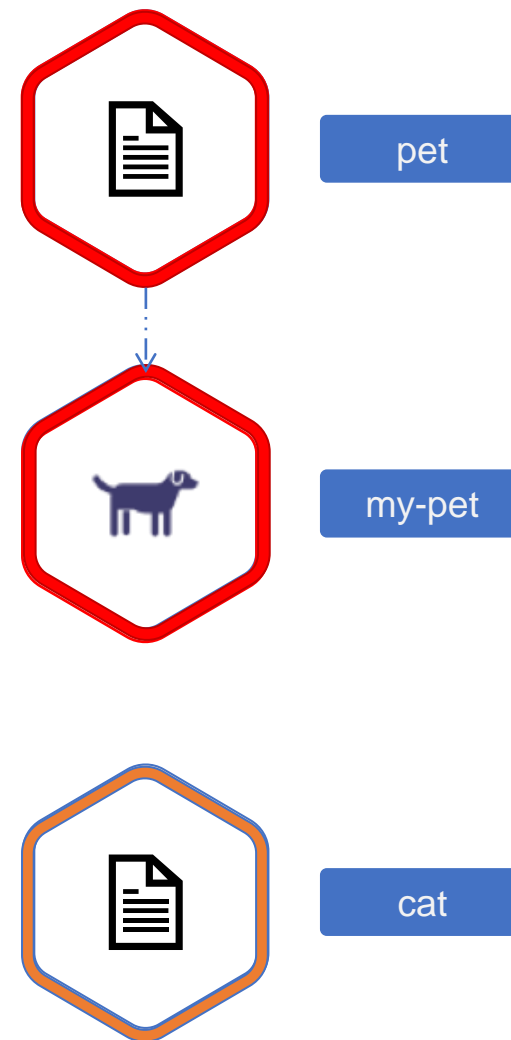
```
local_file.pet: Destroying...
```

```
[id=28b373c6c1fa3fce132a518eadd0175c98f37f20]
```

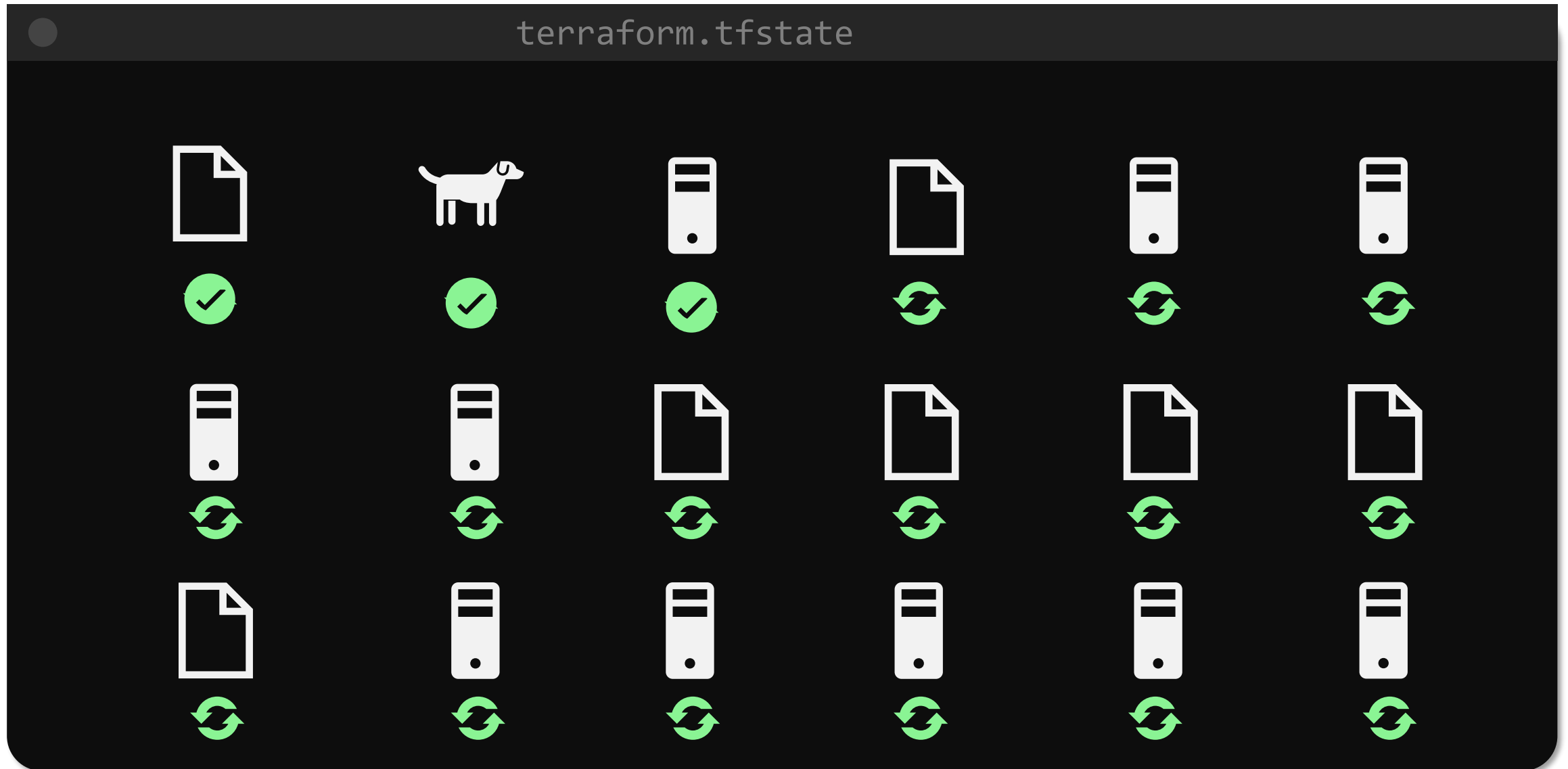
```
local_file.pet: Destruction complete after 0s
```

```
random_pet.my-pet: Destroying... [id=yak]
```

```
random_pet.my-pet: Destruction complete after 0s
```



Performance



Performance

terraform.tfstate

```
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 4,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            ...

```

>_

```
$ terraform plan --refresh=false
```

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

-/+ destroy and then create replacement

Terraform will perform the following actions:

```
    # local_file.cat must be replaced
-/+ resource "local_file" "pet" {
    ~ content           = "I like cats too!" ->
  "Dogs are awesome!" # forces replacement
    directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/root/pets.txt"
    ~ id                 =
  "cba595b7d9f94ba1107a46f3f731912d95fb3d2c" -> (known
after apply)
}
```

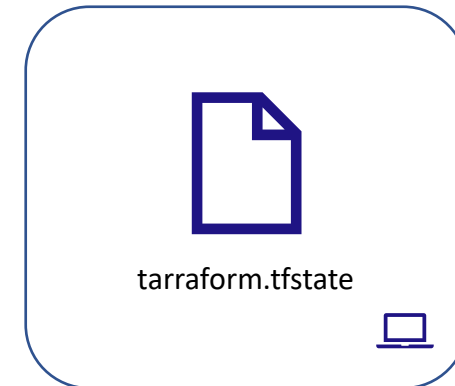
Plan: 1 to add, 0 to change, 1 to destroy.

Collaboration

```
terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 4,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            ...

```

```
> _
$ ls
main.tf variables.tf terraform.tfstate
```



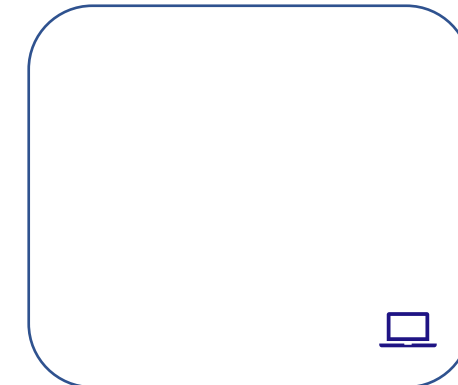
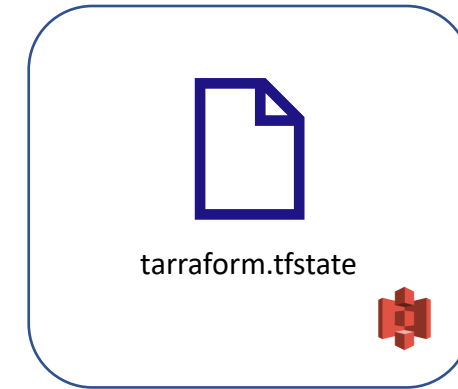
Collaboration

AWS S3

HashiCorp Consul

Google Cloud
Storage

Terraform Cloud



```
terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 4,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            ...
          }
        }
      ]
    }
  ]
}
```



{K}ODE{K}LOUD

The background of the slide features a series of concentric, semi-transparent blue hexagons centered around the text. The hexagons vary in opacity, creating a layered effect. The overall color palette is shades of blue.

Terraform State Considerations

Sensitive Data

```
terraform.tfstate
{
  "mode": "managed",
  "type": "aws_instance",
  "name": "dev-ec2",
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
  "instances": [
    {
      "schema_version": 1,
      "attributes": {
        "ami": "ami-0a634ae95e11c6f91",
        .
        .
        .
        "primary_network_interface_id": "eni-0ccd57b1597e633e0",
        "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
        "private_ip": "172.31.7.21",
        "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
        "public_ip": "54.71.34.19",
        "root_block_device": [
          {
            "delete_on_termination": true,
            "device_name": "/dev/sda1",
            "encrypted": false,
            "iops": 100,
            "kms_key_id": ""
```

Terraform State Considerations

Remote State Backends



Version Control



terraform.tfstate

```
{
  "mode": "managed",
  "type": "aws_instance",
  "name": "dev-ec2",
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
  "instances": [
    {
      "schema_version": 1,
      "attributes": {
        "ami": "ami-0a634ae95e11c6f91",
        .
        .
        .
        "primary_network_interface_id": "eni-0ccd57b1597e633e0",
        "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
        "private_ip": "172.31.7.21",
        "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
        "public_ip": "54.71.34.19",
        "root_block_device": [
          {
            "delete_on_termination": true,
            "device_name": "/dev/sda1",
            "encrypted": false,
            "iops": 100,
            "kms_key_id": "",
            "volume_id": "vol-070720a3636979c22",
            "volume_size": 8,
```

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pet.txt"
  content  = "My favorite pet is Mr.Whiskers!"
}
resource "random_pet" "my-pet" {
  length = 1
}
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```

No Manual Edits

```
terraform.tfstate
{
  "mode": "managed",
  "type": "aws_instance",
  "name": "dev-ec2",
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
  "instances": [
    {
      "schema_version": 1,
      "attributes": {
        "ami": "ami-0a634ae95e11c6f91",
        .
        .
        .
        "primary_network_interface_id": "eni-0ccd57b1597e633e0",
        "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
        "private_ip": "172.31.7.21",
        "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
        "public_ip": "54.71.34.19",
        "root_block_device": [
          {
            "delete_on_termination": true,
            "device_name": "/dev/sda1",
            "encrypted": false,
            "iops": 100,
            "kms_key_id": ""
```



Terraform Commands

terraform validate

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
  file_permissions = "0700"  
}
```

> _

```
$ terraform validate
```

```
Success! The configuration is valid.
```

```
$ terraform validate
```

```
Error: Unsupported argument
```

```
on main.tf line 4, in resource "local_file" "pet":  
  4:     file_permissions = "0777"
```

```
An argument named "file_permissions" is not expected  
here. Did you mean "file_permission"?
```

terraform fmt

main.tf

```
resource "local_file" "pet" {  
  filename      = "/root/pets.txt"  
  content       = "We love pets!"  
  file_permission = "0700"  
}
```

> _

```
$ terraform fmt  
main.tf
```


terraform show

> _

```
$ terraform show
```

```
# local_file.pet:
resource "local_file" "pet" {
  content            = "We love pets!"
  directory_permission = "0777"
  file_permission    = "0777"
  filename           = "/root/pets.txt"
  id                 =
"cba595b7d9f94ba1107a46f3f731912d95fb3d2c"
}
```

> _

```
$ terraform show -json
```

```
{"format_version":"0.1","terraform_version":"0.13.0",
"values":{"root_module":{"resources":[{"address":
"local_file.pet","mode":"managed","type":"local_fil
e","name":"pet","provider_name":"registry.terraform
.io/hashicorp/local","schema_version":0,"values":{"
content":"We love
pets!","content_base64":null,"directory_permission"
:"0777","file_permission":"0777","filename":"/root/
pets.txt","id":"cba595b7d9f94ba1107a46f3f731912d95f
b3d2c","sensitive_content":null}}]}}}
```

terraform providers

main.tf

```
resource "local_file" "pet" {  
  filename      = "/root/pets.txt"  
  content       = "We love pets!"  
  file_permission = "0700"  
}
```

>_

```
$ terraform providers
```

Providers required by configuration:

```
└─ provider[registry.terraform.io/hashicorp/local]
```

Providers required by state:

```
provider[registry.terraform.io/hashicorp/local]
```

```
$ terraform providers mirror /root/terraform/new_local_file
```

- Mirroring hashicorp/local...
- Selected v1.4.0 with no constraints
- Downloading package for windows_amd64...
- Package authenticated: signed by HashiCorp

terraform output

main.tf

```
resource "local_file" "pet" {
  filename      = "/root/pets.txt"
  content       = "We love pets!"
  file_permission = "0777"
}
resource "random_pet" "cat" {
  length      = "2"
  separator   = "-"
}
output content {
  value       = local_file.pet.content
  sensitive   = false
  description = "Print the content of the file"
}
output pet-name {
  value       = random_pet.cat.id
  sensitive   = false
  description = "Print the name of the pet"
}
```

>_

```
$ terraform output
```

```
content = We love pets!
pet-name = huge-owl
```

```
$ terraform output pet-name
```

```
pet-name = huge-owl
```

terraform refresh

main.tf

```
resource "local_file" "pet" {  
  filename      = "/root/pets.txt"  
  content       = "We love pets!"  
  file_permission = "0777"  
}  
resource "random_pet" "cat" {  
  length      = "2"  
  separator   = "-"  
}
```

>_

\$ terraform refresh

```
random_pet.cat: Refreshing state... [id=huge-owl]  
local_file.pet: Refreshing state...  
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]
```

\$ terraform plan

```
Refreshing Terraform state in-memory prior to plan...  
The refreshed state will be used to calculate this  
plan, but will not be  
persisted to local or remote state storage.
```

```
random_pet.cat: Refreshing state... [id=huge-owl]  
local_file.pet: Refreshing state...  
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]
```

No changes. Infrastructure is up-to-date.

terraform graph

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content  = "My favorite pet is ${random_pet.m
y-pet.id}"
}
resource "random_pet" "my-pet" {
  prefix = "Mr"
  separator = "."
  length = "1"
}
```

> _

```
$ terraform graph

digraph {
  compound = "true"
  newrank = "true"
  subgraph "root" {
    "[root] local_file.pet (expand)" [label =
"local_file.pet", shape = "box"]
    "[root]
provider[\"registry.terraform.io/hashicorp/local\"]" [label =
"provider[\"registry.terraform.io/hashicorp/local\"]", shape =
"diamond"]
    "[root]
provider[\"registry.terraform.io/hashicorp/random\"]" [label =
"provider[\"registry.terraform.io/hashicorp/random\"]", shape =
"diamond"]
    "[root] random_pet.my-pet (expand)" [label =
"random_pet.my-pet", shape = "box"]
    "[root] local_file.pet (expand)" -> "[root]
provider[\"registry.terraform.io/hashicorp/local\"]"
    "[root] local_file.pet (expand)" -> "[root]
random_pet.my-pet (expand)"
    "[root] meta.count-boundary (EachMode fixup)" -
> "[root] local_file.pet (expand)"
    "[root]
provider[\"registry.terraform.io/hashicorp/local\"] (close)" ->
"[root] local_file.pet (expand)"
  }
```

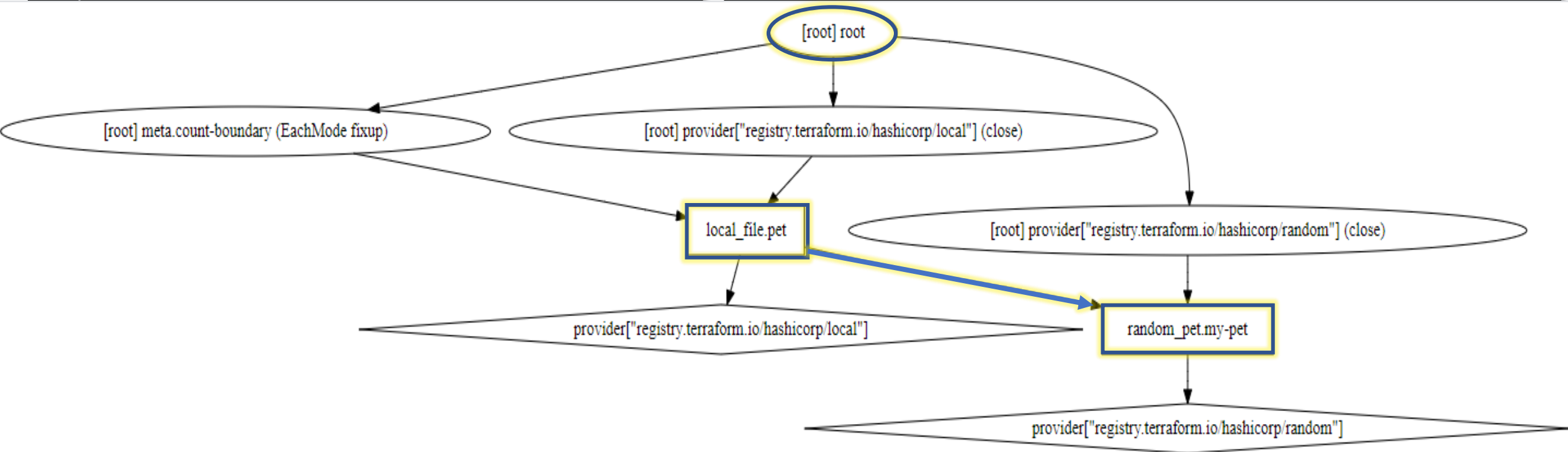
terraform graph

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content  = "My favorite pet is ${random_pet.m  
y-pet.id}"  
}
```

> _

```
$ apt update  
$ apt install graphviz -y  
$ terraform graph | dot -Tsvg > graph.svg
```





Mutable vs Immutable Infrastructure

terraform validate

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
  file_permission = "0700"  
}
```



> _

```
$ terraform apply
```

```
#local_file.pet must be replaced
```

```
-/+ resource "local_file" "pet" {  
      content                = "We love pets!"  
      directory_permission   = "0777"  
      ~ file_permission       = "0777" -> "0700" # forces  
replacement  
      filename                = "/root/pet.txt"  
      ~ id                    =  
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after  
apply)  
}
```

```
Plan: 1 to add, 0 to change, 1 to destroy.
```

```
Do you want to perform these actions?
```

```
Terraform will perform the actions described above.  
Only 'yes' will be accepted to approve.
```

```
Enter a value: yes
```

```
local_file.pet: Destroying...
```

```
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
```

```
local_file.pet: Destruction complete after 0s
```

```
local_file.pet: Creating...
```



Configuration Drift





v1.17



v1.18



v1.18



v1.18



Immutable Infrastructure



Immutable Infrastructure



Immutable Infrastructure

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
  file_permission = "0700"  
}
```



>_

```
$ terraform apply  
  
#local_file.pet must be replaced  
-/+ resource "local_file" "pet" {  
    content                = "We love pets!"  
    directory_permission = "0777"  
    ~ file_permission      = "0777" -> "0700" # forces  
replacement  
    filename                = "/root/pet.txt"  
    ~ id                    =  
    "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after  
apply)  
}
```

Plan: 1 to add, 0 to change, 1 to destroy.

```
local_file.pet: Destroying...  
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]  
local_file.pet: Destruction complete after 0s  
local_file.pet: Creating...  
local_file.pet: Creation complete after 0s  
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
```

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.



The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in varying shades of blue, creating a tunnel-like effect. In the top-left and bottom-right corners, there are additional geometric shapes, including triangles and polygons, also in shades of blue.

Lifecycle Rules

create_before_destroy

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
  file_permission = "0700"  
  
  lifecycle {  
    create_before_destroy = true  
  }  
}
```



>_

```
$ terraform apply
```

```
#local_file.pet must be replaced
```

```
-/+ resource "local_file" "pet" {  
    content                = "We love pets!"  
    directory_permission   = "0777"  
    ~ file_permission      = "0777" -> "0755" # forces repl  
    filename               = "/root/pet.txt"  
    ~ id                   =  
    "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after ap  
    }  
}
```

```
Plan: 1 to add, 0 to change, 1 to destroy.
```

```
...
```

```
local_file.pet: Creating...
```

```
local_file.pet: Creation complete after 0s
```

```
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
```

```
local_file.pet: Destroying...
```

```
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
```

```
local_file.pet: Destruction complete after 0s
```

```
Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

prevent_destroy

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
  file_permission = "0700"  
  
  lifecycle {  
    prevent_destroy = true  
  }  
  
}
```



>_

```
$ terraform apply  
local_file.my-pet: Refreshing state...  
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]
```

Error: Instance cannot be destroyed

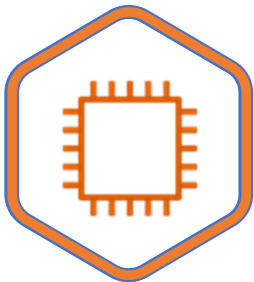
```
on main.tf line 1:  
1: resource "local_file" "my-pet" {
```

Resource local_file.my-pet has lifecycle.prevent_destroy set, but the plan calls for this resource to be destroyed. To avoid this error and continue with the plan, either disable lifecycle.prevent_destroy or reduce the scope of the plan using the -target flag.

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {  
  ami          = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "ProjectA-Webserver"  
  }  
}
```



>_

```
$ terraform apply
```

```
...
```

Terraform will perform the following actions:

```
# aws_instance.webserver will be created  
+ resource "aws_instance" "webserver" {  
  + ami          = "ami-0edab43b6fa892279"  
  + get_password_data = false  
  + host_id       = (known after apply)  
  + id            = (known after apply)  
  + instance_state = (known after apply)  
  + instance_type = "t2.micro"  
  + tags         {  
    + "Name" = "ProjectA-WebServer"  
  }  
}
```

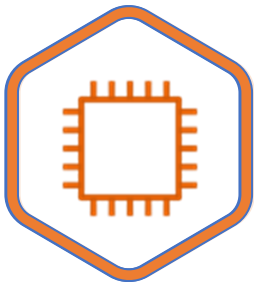
```
.  
aws_instance.webserver: Creation complete after 33s [id=i-  
05cd83b221911acd5]
```

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {  
  ami           = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "ProjectA-Webserver"  
  }  
}
```



>_

```
$ terraform apply
```

```
aws_instance.webserver: Refreshing state... [id=i-  
05cd83b221911acd5]
```

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:

~ update in-place

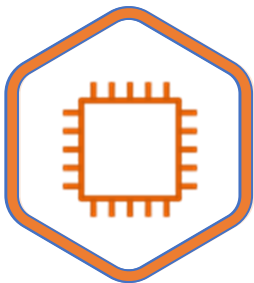
Terraform will perform the following actions:

```
# aws_instance.webserver will be updated in-place  
~ resource "aws_instance" "webserver" {  
  .  
  .  
  ~ tags = {  
    ~ "Name" = "ProjectB-WebServer" -> "ProjectA-WebServer"  
  }  
  .  
  .  
Apply complete! Resources: 0 added, 1 changed, 0 destroyed.
```

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {  
  ami           = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "ProjectA-Webserver"  
  }  
  lifecycle {  
    ignore_changes = [  
      tags  
    ]  
  }  
}
```



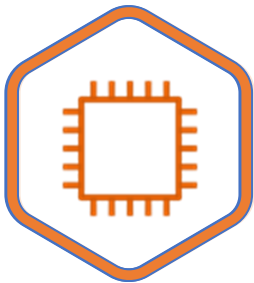
>_

```
$ terraform apply  
aws_instance.webserver: Refreshing state... [id=i-  
05cd83b221911acd5]  
  
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {  
  ami          = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "ProjectA-Webserver"  
  }  
  lifecycle {  
    ignore_changes = all  
  }  
}
```



>_

```
$ terraform apply
```

```
aws_instance.webserver: Refreshing state... [id=i-  
05cd83b221911acd5]
```

```
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```


Order	Option	
1	create_before_destroy	Create the resource first and then destroy older
2	prevent_destroy	Prevents destroy of a resource
3	ignore_changes	Ignore Changes to Resource Attributes (specific/all)

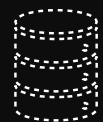


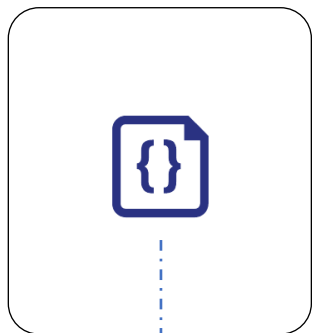
{K}ODE{K}LOUD

Data Sources



Real World Infrastructure

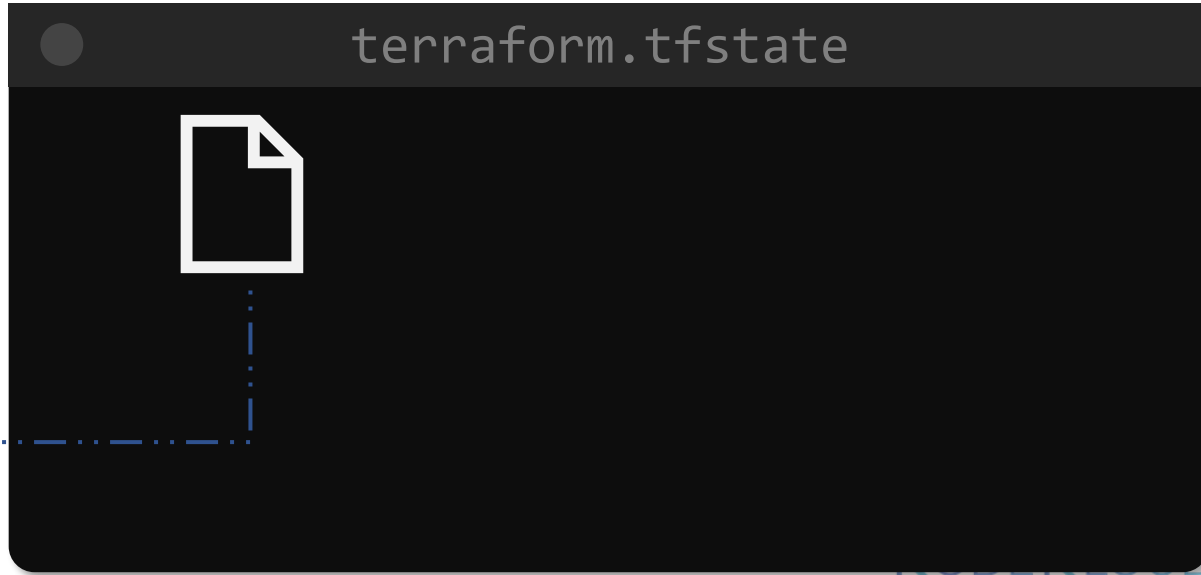
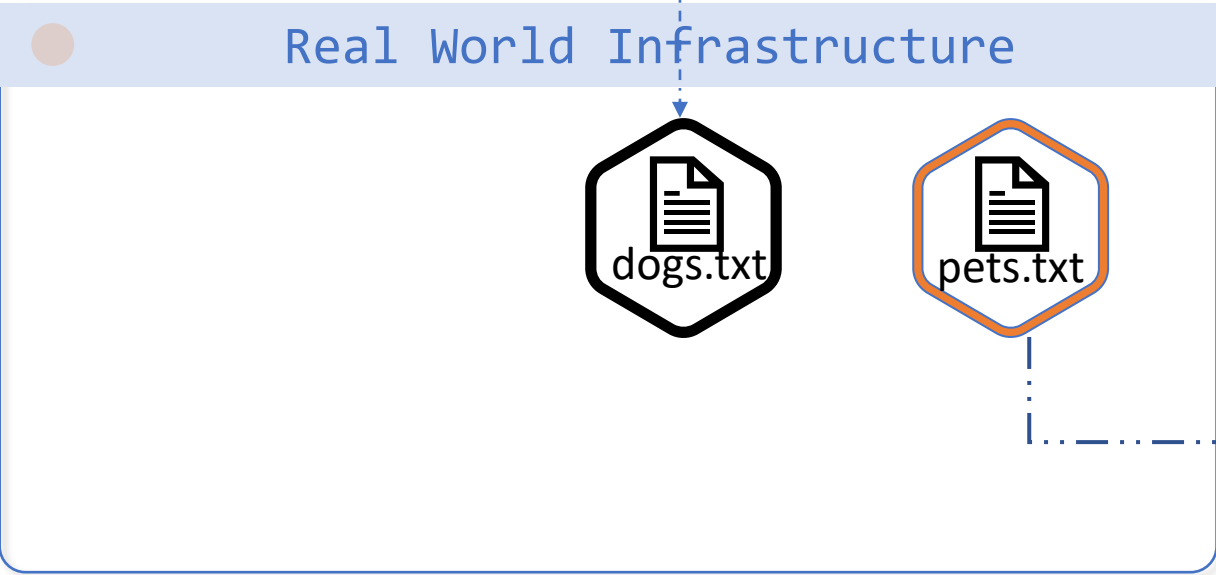




```
main.tf

resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```

```
>_
$ cat /root/dog.txt
Dogs are awesome!
```



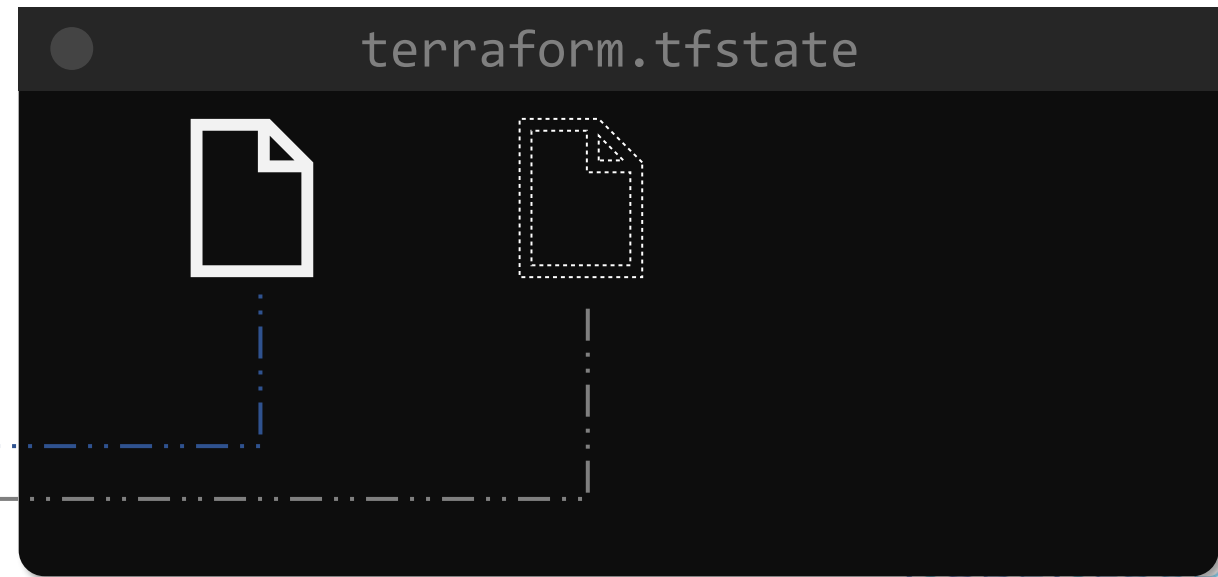
Data Sources



```
> _  
$ cat /root/dog.txt  
Dogs are awesome!
```

```
main.tf  
  
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = data.local_file.dog.content  
}  
  
data "local_file" "dog" {  
  filename = "/root/dog.txt"  
}
```

Real World Infrastructure



LOCAL DOCUMENTATION

Filter

local provider

▼ Resources

local_file

▼ Data Sources

• local_file

Argument Reference

The following argument is required:

- `filename` - (Required) The path to the file that will be read. The data source will return an error if the file does not exist.

Attributes Exported

The following attribute is exported:

- `content` - The raw content of the file that was read.
- `content_base64` - The base64 encoded version of the file content (use this when dealing with binary data).



Resource	Data Source
Keyword: resource	Keyword: data
Creates, Updates, Destroys Infrastructure	Only Reads Infrastructure
Also called Managed Resources	Also called Data Resources



{K}ODE{K}LOUD

Meta Arguments

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  content = var.content  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}  
variable "content" {  
  default = "I love pets!"  
}
```



Shell Scripts

create_files.sh

```
#!/bin/bash
```

```
for i in {1..3}
do
    touch /root/pet${i}
done
```

>_

```
$ ls -ltr /root/
```

```
-rw-r--r-- 1 root root 0 Sep  9 02:04 pet2
-rw-r--r-- 1 root root 0 Sep  9 02:04 pet1
-rw-r--r-- 1 root root 0 Sep  9 02:04 pet3
```

Iteration	filename
1	/root/pet1
2	/root/pet2
3	/root/pet3

Meta Arguments

depends_on

```
main.tf

resource "local_file" "pet" {
  filename = var.filename
  content  = var.content
  depends_on = [
    random_pet.my-pet
  ]
}

resource "random_pet" "my-pet" {
  prefix      = var.prefix
  separator   = var.separator
  length      = var.length
}
```

lifecycle

```
main.tf

resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content  = "We love pets!"
  file_permission = "0700"
  lifecycle {
    create_before_destroy = true
  }
}
```



The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in varying shades of blue, creating a tunnel-like effect. In the top-left and bottom-right corners, there are faint, light-blue geometric shapes that resemble stylized letters or symbols.

Count

count

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  
  count = 3  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}
```

>_

```
$ terraform plan
```

```
[Output Truncated]
```

```
Terraform will perform the following actions:
```

```
...
```

```
# local_file.pet[2] will be created
```

```
+ resource "local_file" "pet" {  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename              = "/root/pets.txt"  
  + id                   = (known after apply)  
}
```

```
Plan: 3 to add, 0 to change, 0 to destroy.
```



count

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  
  count    = 3  
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}
```

>_

```
$ terraform apply
```

```
[Output Truncated]
```

```
.
```

```
local_file.pet[2]: Creating...
```

```
local_file.pet[0]: Creating...
```

```
local_file.pet[1]: Creating...
```

```
local_file.pet[0]: Creation complete after 0s
```

```
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]
```

```
local_file.pet[2]: Creation complete after 0s
```

```
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]
```

```
local_file.pet[1]: Creation complete after 0s
```

```
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]
```

```
Apply complete! Resources: 3 added, 0 changed, 0  
destroyed
```

count

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename  
  count    = 3  
}
```

variables.tf

```
variable "filename" {  
  default = "/root/pets.txt"  
}
```

>_

```
$ ls /root  
pet.txt
```

pet[0]



pet[1]



pet[2]



count

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename[count.index]  
  count    = 3  
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {  
  default = [  
    "/root/pets.txt",  
    "/root/dogs.txt",  
    "/root/cats.txt"  
  ]  
}
```

> _

```
$ ls /root
```

```
pets.txt  
dogs.txt  
cats.txt
```

Length Function

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename[count.index]  
  count    = length(var.filename)  
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {  
  default = [  
    "/root/pets.txt",  
    "/root/dogs.txt",  
    "/root/cats.txt",  
    "/root/cows.txt",  
    "/root/ducks.txt"  
  ]  
}
```

> _

```
$ ls /root
```

```
pets.txt  
dogs.txt  
cats.txt
```

Length Function

variable	function	value
<code>fruits = ["apple", "banana", "orange"]</code>	<code>length(fruits)</code>	3
<code>cars = ["honda", "bmw", "nissan", "kia"]</code>	<code>length(cars)</code>	4
<code>colors = ["red", "purple"]</code>	<code>length(colors)</code>	2

LengthFunction

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename[count.index]  
  count    = length(var.filename)  
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {  
  default = [  
    "/root/pets.txt",  
    "/root/dogs.txt",  
    "/root/cats.txt",  
    "/root/cows.txt",  
    "/root/ducks.txt"  
  ]  
}
```

> _

```
$ ls /root
```

```
pets.txt  
dogs.txt  
cats.txt
```

```
> _  
  
$ terraform apply  
  
.  
.  
Terraform will perform the following actions:  
  
# local_file.pet[0] will be created  
+ resource "local_file" "pet" {  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename              = "/root/pets.txt"  
  + id                   = (known after apply)  
}  
  
# local_file.pet[1] will be created  
+ resource "local_file" "pet" {  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename              = "/root/dogs.txt"  
  + id                   = (known after apply)  
}  
  
# local_file.pet[2] will be created  
+ resource "local_file" "pet" {  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename              = "/root/cats.txt"  
  + id                   = (known after apply)  
}
```

```
> _  
  
$ ls /root  
pet.txt  
dogs.txt  
cats.txt
```

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename[count.index]  
  count    = length(var.filename)  
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {  
  default = [  
    "/root/dogs.txt",  
    "/root/cats.txt"  
  ]  
}
```


main.tf

```
resource "local_file" "pet" {
  filename = var.filename[count.index]
  count    = length(var.filename)
}
```

pet[0]



Replace

pet[1]



Replace

pet[2]



Destroy

variables.tf

```
variable "filename" {
  default = [
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```

> _

\$ terraform plan

```
...
# local_file.pet[0] must be replaced
-/+ resource "local_file" "pet" {
  directory_permission = "0777"
  file_permission      = "0777"
  ~ filename           = "/root/pets.txt" -> "/root/dogs.txt" #
forces replacement
}
# local_file.pet[1] must be replaced
-/+ resource "local_file" "pet" {
  directory_permission = "0777"
  file_permission      = "0777"
  ~ filename           = "/root/dogs.txt" -> "/root/cats.txt" #
forces replacement
}
# local_file.pet[2] will be destroyed
- resource "local_file" "pet" {
  - directory_permission = "0777" -> null
  - file_permission      = "0777" -> null
}
```

main.tf

```
resource "local_file" "pet" {  
  filename = var.filename[count.index]  
  count    = length(var.filename)  
}  
  
output "pets" {  
  value = local_file.pet  
}
```

pet[0]



pet[1]



pet[2]

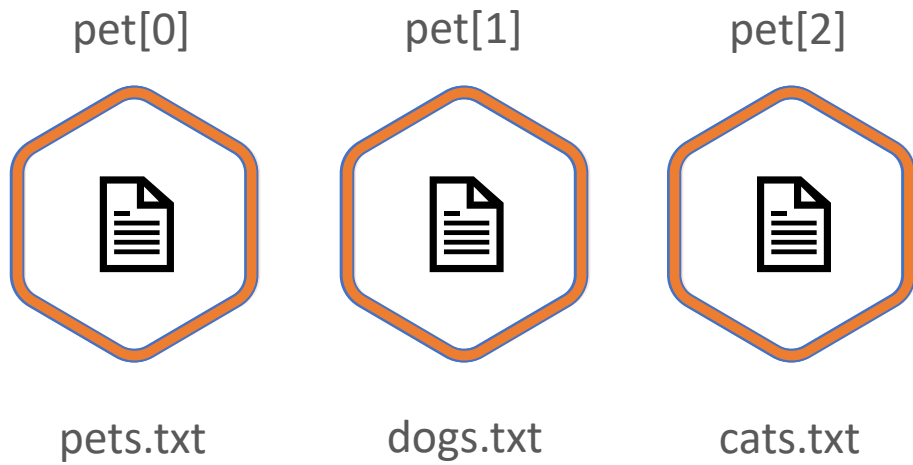


>_

\$ terraform output

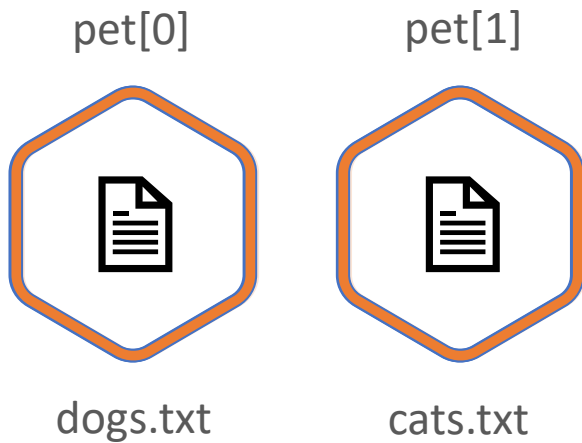
Outputs:

```
pets = [  
  {  
    "directory_permission" = "0777"  
    "file_permission"      = "0777"  
    "filename"             = "/root/pets.txt"  
    "id"                   = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  },  
  {  
    "directory_permission" = "0777"  
    "file_permission"      = "0777"  
    "filename"             = "/root/dogs.txt"  
    "id"                   = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  },  
  {  
    "directory_permission" = "0777"  
    "file_permission"      = "0777"  
    "filename"             = "/root/cats.txt"  
    "id"                   = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  },  
]
```



```
variables.tf

variable "filename" {
  default = [
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```



```
variables.tf

variable "filename" {
  default = [
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```

Resource	Resource Updates	Action
pet[0]	/root/pets.txt" -> "/root/dogs.txt"	Destroy and Replace
pet[1]	"/root/dogs.txt" -> "/root/cats.txt"	Destroy and Replace
pet[2]	Does not Exist	Destroy



{K}ODE{K}LOUD

The background of the slide is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in a lighter shade of blue, creating a tunnel-like effect. The text 'for_each' is centered within these hexagons.

`for_each`

for_each

main.tf

```
resource "local_file" "pet" {  
  filename = each.value  
  for_each = var.filename  
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {  
  type = string  
  default = [  
    "/root/pets.txt",  
    "/root/dogs.txt",  
    "/root/cats.txt"  
  ]  
}
```

>_

\$ terraform plan

```
Terraform will perform the following actions:  
# local_file.pet["/root/cats.txt"] will be created  
+ resource "local_file" "pet" {  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename              = "/root/cats.txt"  
}  
... <output trimmed>  
Plan: 3 to add, 0 to change, 0 to destroy.
```

for_each

main.tf

```
resource "local_file" "pet" {  
  filename = each.value  
  for_each = toset(var.filename)  
}
```

variables.tf

```
variable "filename" {  
  type = string  
  default = [  
    "/root/pets.txt",  
    "/root/dogs.txt",  
    "/root/cats.txt"  
  ]  
}
```

pet[0]



pet[1]



pet[2]



>_

\$ terraform plan

```
Terraform will perform the following actions:  
# local_file.pet["/root/cats.txt"] will be created  
+ resource "local_file" "pet" {  
  + directory_permission = "0777"  
  + file_permission      = "0777"  
  + filename              = "/root/cats.txt"  
}  
... <output trimmed>  
Plan: 3 to add, 0 to change, 0 to destroy.
```


for_each

main.tf

```
resource "local_file" "pet" {
  filename = each.value
  for_each = toset(var.filename)
}

output "pets" {
  value = local_file.pet
}
```

pet[0]



pet[1]



pet[2]



variables.tf

```
variable "filename" {
  type = list(string)
  default = [
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```

>_

\$ terraform plan

Terraform will perform the following actions:

```
# local_file.pet["/root/pets.txt"] will be destroyed
+ resource "local_file" "pet" {
  + directory_permission = "0777"
  + file_permission      = "0777"
  + filename              = "/root/pets.txt"
}
```

... <output trimmed>

Plan: 0 to add, 0 to change, 1 to destroy.

for_each

main.tf

```
resource "local_file" "pet" {  
  filename = each.value  
  for_each = toset(var.filename)  
}  
  
output "pets" {  
  value = local_file.pet  
}
```

pet[0]



pet[1]



pet[2]



>_

```
$ terraform output  
  
pets = {  
  "/root/cats.txt" = {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/cats.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  }  
  
  "/root/dogs.txt" = {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/dogs.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  }  
}
```

count

for_each

```
>_  
$ terraform output  
pets = [  
  {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/pets.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  },  
  {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/dogs.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  },  
  {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/cats.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  },  
]
```

```
>_  
$ terraform output  
pets = {  
  "/root/cats.txt" = {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/cats.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  }  
  
  "/root/dogs.txt" = {  
    "directory_permission" = "0777"  
    "file_permission" = "0777"  
    "filename" = "/root/dogs.txt"  
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"  
  }  
}
```



{K}ODE{K}LOUD

Version Constraints

main.tf

```
resource "local_file" "pet" {  
  filename    = "/root/pet.txt"  
  content     = "We love pets!"  
}
```

>_

```
$ terraform init
```

Initializing the backend...

Initializing provider plugins...

- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration, so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking changes, we recommend adding version constraints in a `required_providers` block in your configuration, with the constraint strings suggested below.

```
* hashicorp/local: version = "~> 1.4.0"
```

Terraform has been successfully initialized!

main.tf

```
resource "local_file" "pet" {  
  filename    = "/root/pet.txt"  
  content     = "We love pets!"  
}
```



Terraform | Registry

Search Providers and Modules

Providers

/ hashicorp

/ local

Version 2.0.0

Latest Version

local



local



Official

by: HashiCorp

Utility

Used to manage local resources, such as creating files

VERSION

2.0.0

PUBLISHED

9 days ago

INSTALLS

15.8M

SOURCE CODE

[hashicorp/terraform-provider-local](https://github.com/hashicorp/terraform-provider-local)

main.tf

```
resource "local_file" "pet" {  
  filename    = "/root/pet.txt"  
  content     = "We love pets!"  
}
```

The screenshot shows the Terraform Registry page for the 'local' provider. The page has a purple header with the Terraform logo and 'Registry' text. A search bar is on the right. Below the header, there are breadcrumbs: 'Providers / hashicorp / local / Version 2.0.0'. A dropdown menu is open for 'Version 2.0.0', showing a list of versions: 'Version 2.0.0' (marked as the latest version with a checkmark), 'Version 1.4.0', 'Version 1.3.0', 'Version 1.2.2', and 'Version 1.2.1'. The main content area shows the 'local' provider logo, an 'Official' badge, and a 'Utility' badge. The text 'Used to manage' is partially visible at the bottom.

Terraform Registry

Search Providers and Modules

Providers / hashicorp / local / Version 2.0.0 Latest Version

local

LATEST VERSION

Version 2.0.0 ✓
Published 9 days ago

Version 1.4.0
Published a year ago

Version 1.3.0
Published a year ago

Version 1.2.2
Published a year ago

Version 1.2.1

Used to manage

main.tf

```
resource "local_file" "pet" {  
  filename    = "/root/pet.txt"  
  content     = "We love pets!"  
}
```

[Overview](#)

[Documentation](#)

[USE PROVIDER](#) ▼

How to use this provider

To install this provider, copy and paste this code into your Terraform configuration. Then, run `terraform init`.

Terraform 0.13

Latest

```
terraform {  
  required_providers {  
    local = {  
      source = "hashicorp/local"  
      version = "1.4.0"  
    }  
  }  
}
```

main.tf

```
terraform {  
  required_providers {  
    local = {  
      source = "hashicorp/local"  
      version = "1.4.0"  
    }  
  }  
}  
  
resource "local_file" "pet" {  
  filename    = "/root/pet.txt"  
  content     = "We love pets!"  
}
```

[Overview](#)

[Documentation](#)

 [USE PROVIDER](#) ▼

How to use this provider

To install this provider, copy and paste this code into your Terraform configuration. Then, run `terraform init`.

Terraform 0.13

[Latest](#)

```
terraform {  
  required_providers {  
    local = {  
      source = "hashicorp/local"  
      version = "1.4.0"  
    }  
  }  
}
```

main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}

resource "local_file" "pet" {
  filename    = "/root/pet.txt"
  content     = "We love pets!"
}
```

> _

```
$ terraform init
```

Initializing the backend...

Initializing provider plugins...

- Finding hashicorp/local versions matching "1.4.0"...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

main.tf

```
terraform {  
  required_providers {  
    local = {  
      source = "hashicorp/local"  
      version = "> 1.2.0, < 2.0.0, != 1.4.0"  
    }  
  }  
}  
  
resource "local_file" "pet" {  
  filename    = "/root/pet.txt"  
  content     = "We love pets!"  
}
```

>_

```
$ terraform init
```

Initializing the backend...

Initializing provider plugins...

- Finding hashicorp/local versions matching "> 1.2.0, < 2.0.0, != 1.4.0"...
- Installing hashicorp/local v1.3.0...
- Installed hashicorp/local v1.3.0 (signed by HashiCorp)

Terraform has been successfully initialized!

main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "~> 1.2.0"
    }
  }
}

resource "local_file" "pet" {
  filename    = "/root/pet.txt"
  content     = "We love pets!"
}
```

>_

\$ terraform init

Initializing

Initializing

- Finding h

1.2.0"...

- Installin

- Installed

HashiCorp)

Terraform h

The screenshot shows the Terraform Registry page for the 'local' provider. The page has a purple header with the Terraform logo and 'Registry' text. Below the header, there's a breadcrumb trail: 'Providers / hashicorp / local / Version 2.0.0'. A search bar on the right contains the text 'local'. The main content area shows the 'local' provider with a yellow 'Official' badge. To the right, a dropdown menu is open, showing a list of versions: 'Version 2.0.0' (marked as the latest version with a checkmark), 'Version 1.4.0', 'Version 1.3.0', 'Version 1.2.2', and 'Version 1.2.1'. The provider's logo, a stylized 'H', is displayed on the left. Below the logo, the text 'local' is followed by 'Official' and 'by: HashiCorp'. A 'Utility' button is visible, and the text 'Used to manage' is at the bottom.

LATEST VERSION	
Version 2.0.0	Published 21 days ago
Version 1.4.0	Published a year ago
Version 1.3.0	Published a year ago
Version 1.2.2	Published 2 years ago
Version 1.2.1	Published 2 years ago



{K}ODE{K}LOUD

Getting Started with AWS

Why AWS?



Why AWS?

Compute



Databases



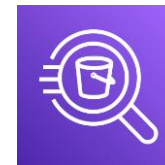
Storage



Machine Learning



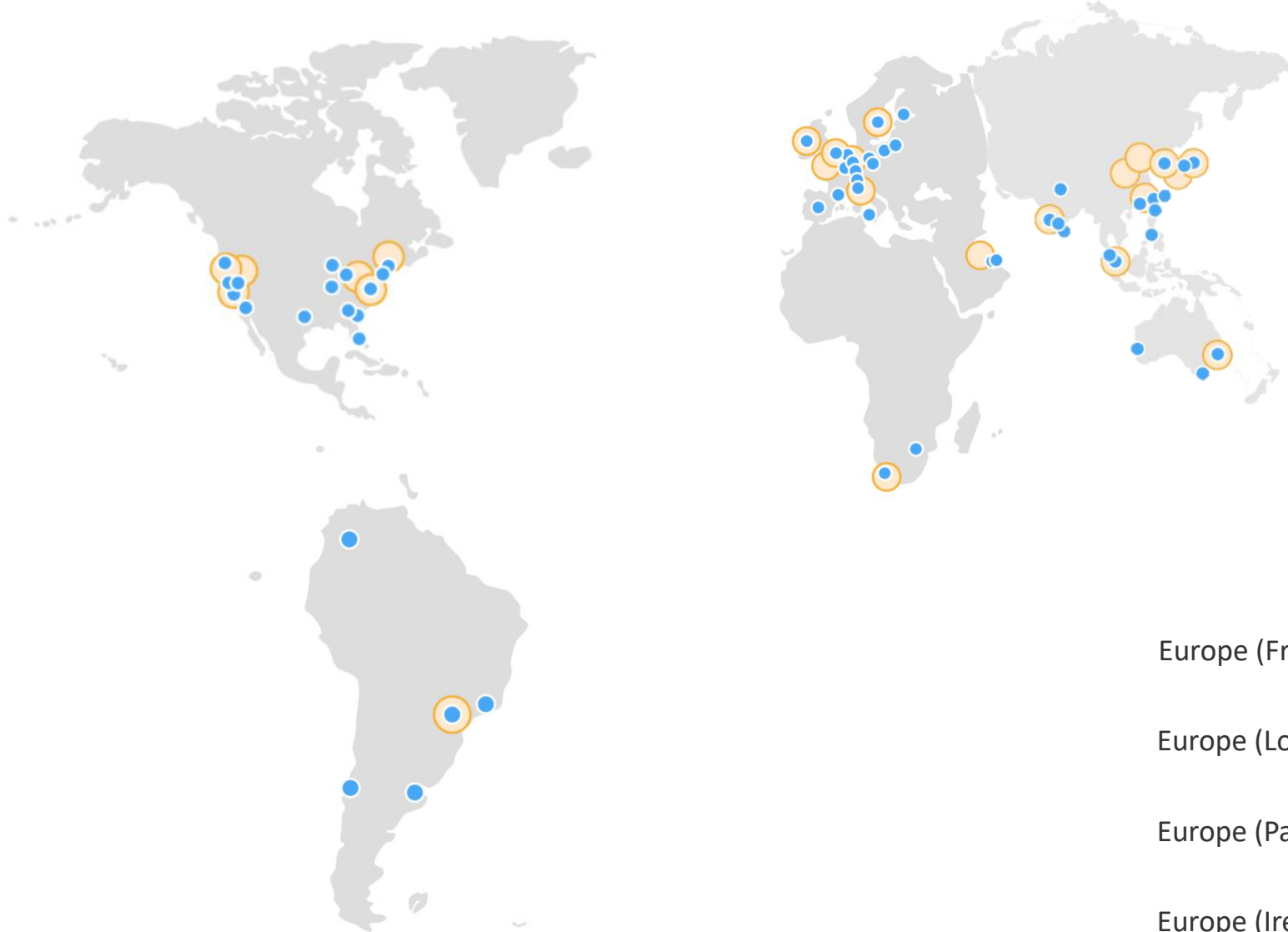
Analytics



IoT



Why AWS?



US East (Ohio) Region

US West (Oregon) Region

US West (Northern California) Region

GovCloud (US-West) Region

GovCloud (US-East) Region

Canada (Central) Region

South America (São Paulo) Region

Europe (Frankfurt) Region

Mainland China (Beijing) Region

Europe (London) Region

Asia Pacific (Sydney) Region

Europe (Paris) Region

Asia Pacific (Tokyo) Region

Europe (Ireland) Region

Asia Pacific (Mumbai) Region

Europe (Milan) Region

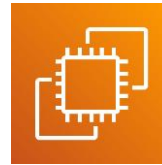
Asia Pacific (Hong Kong) Region

Why AWS with Terraform?

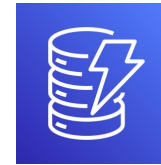


aws

aws



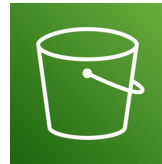
EC2



DynamoDB



Elastic Block Store



Simple
Storage Service (S3)



Route 53



VPC

Getting Started with AWS

Demo: Setup an AWS Account

Introduction to IAM

Demo: IAM

Programmatic Access

IAM with Terraform



Introduction to AWS S3

S3 with Terraform

Introduction to DynamoDB

Demo DynamoDB

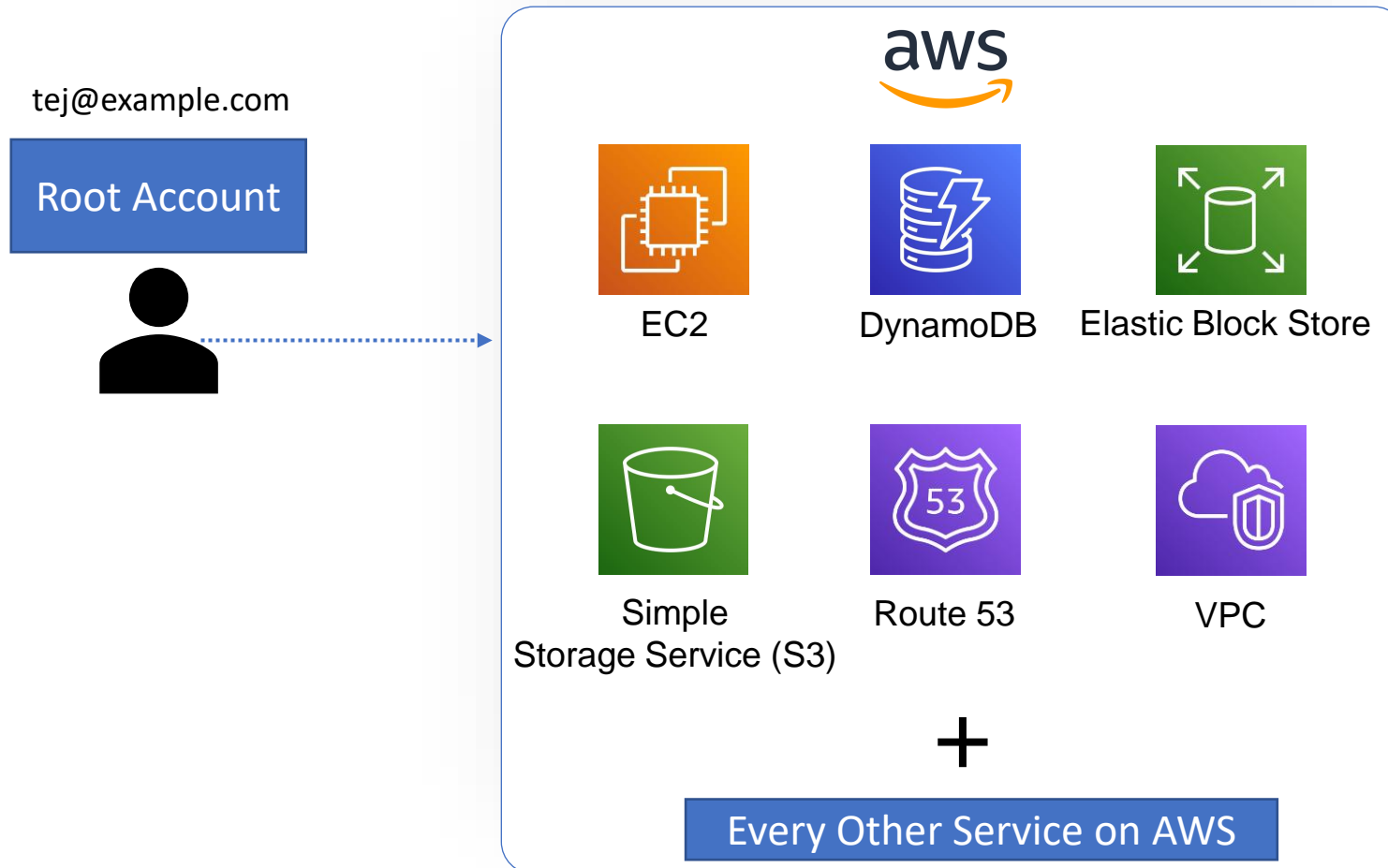
DynamoDB with Terraform





Introduction to IAM

Identity and Access Management in AWS





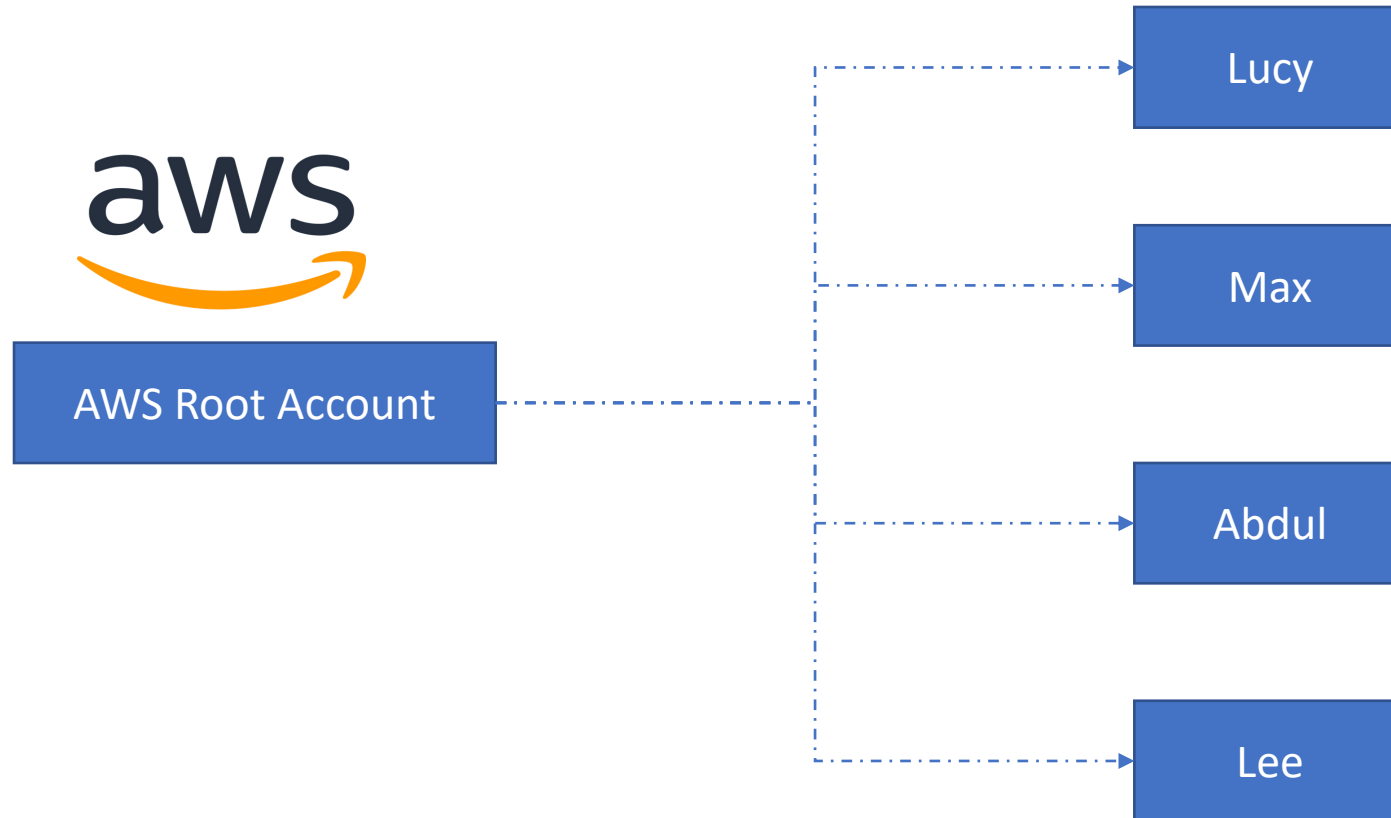
Linux Root User



Windows Admin User

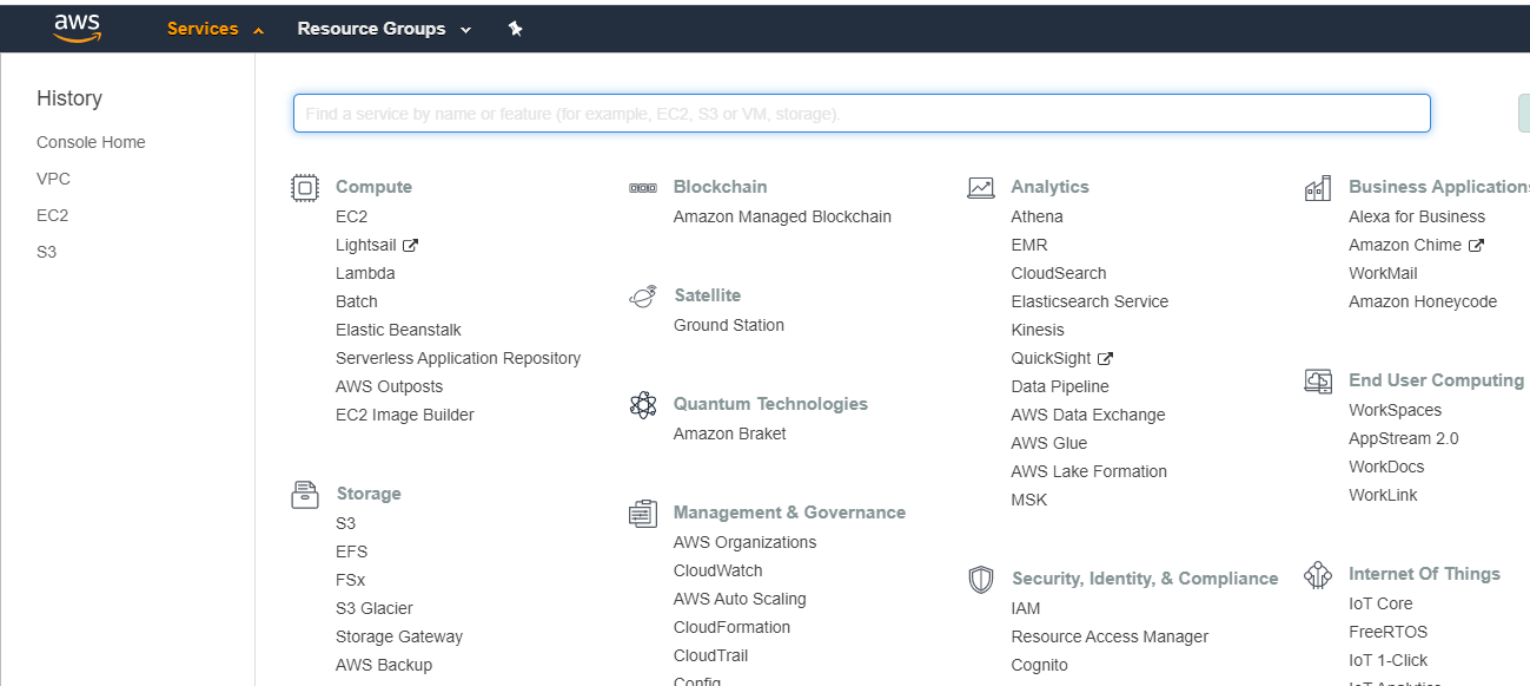


AWS Root Account





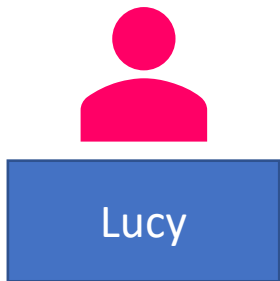
Username
Password



console.aws.com

```
$ aws s3api create-bucket --bucket my-bucket --region us-east-1
```

Access Key ID
Secret Access Key



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

AdministratorAccess
Policy











IAM Policy

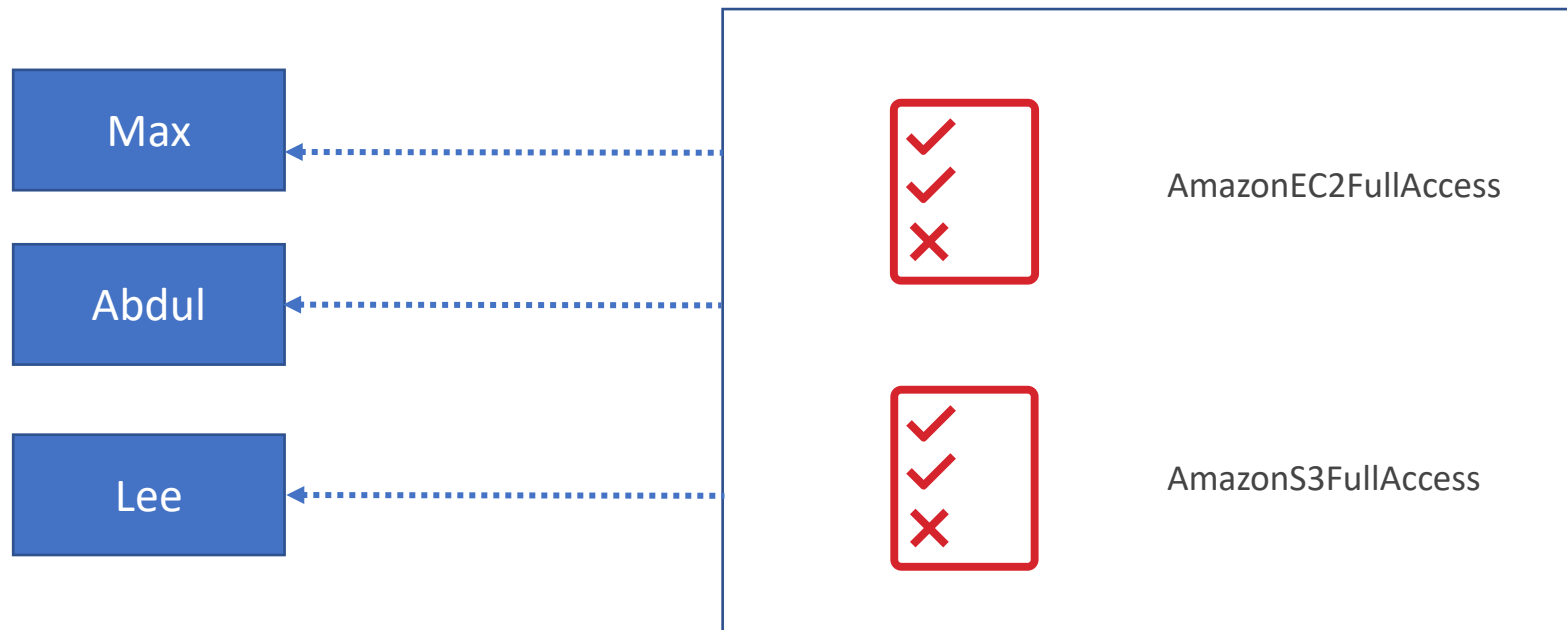
AdministratorAccess

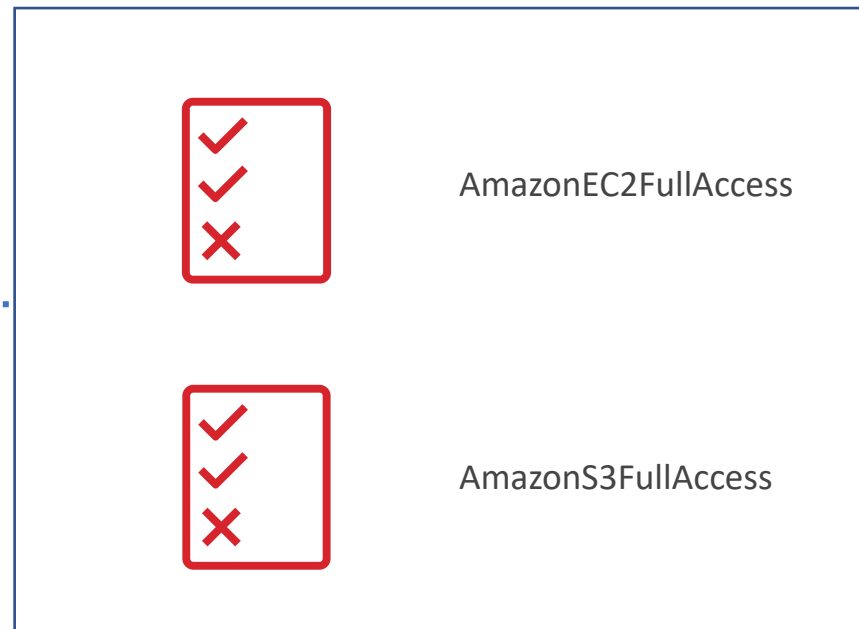
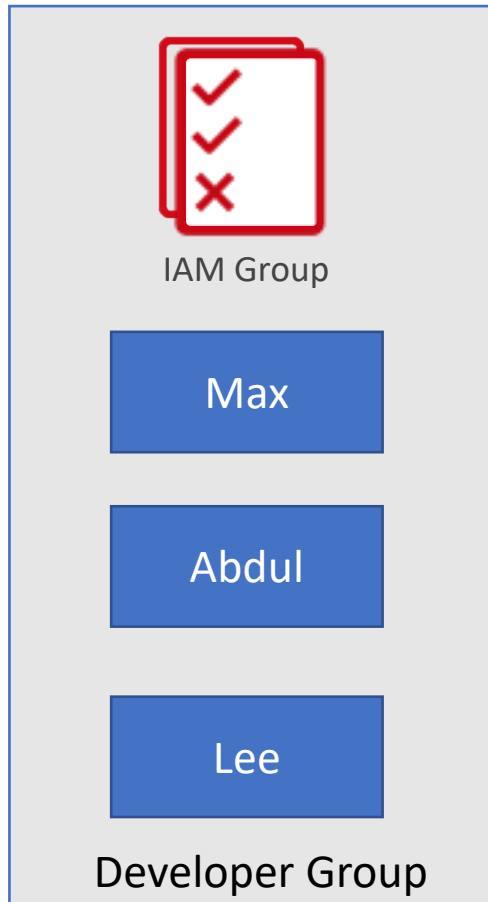


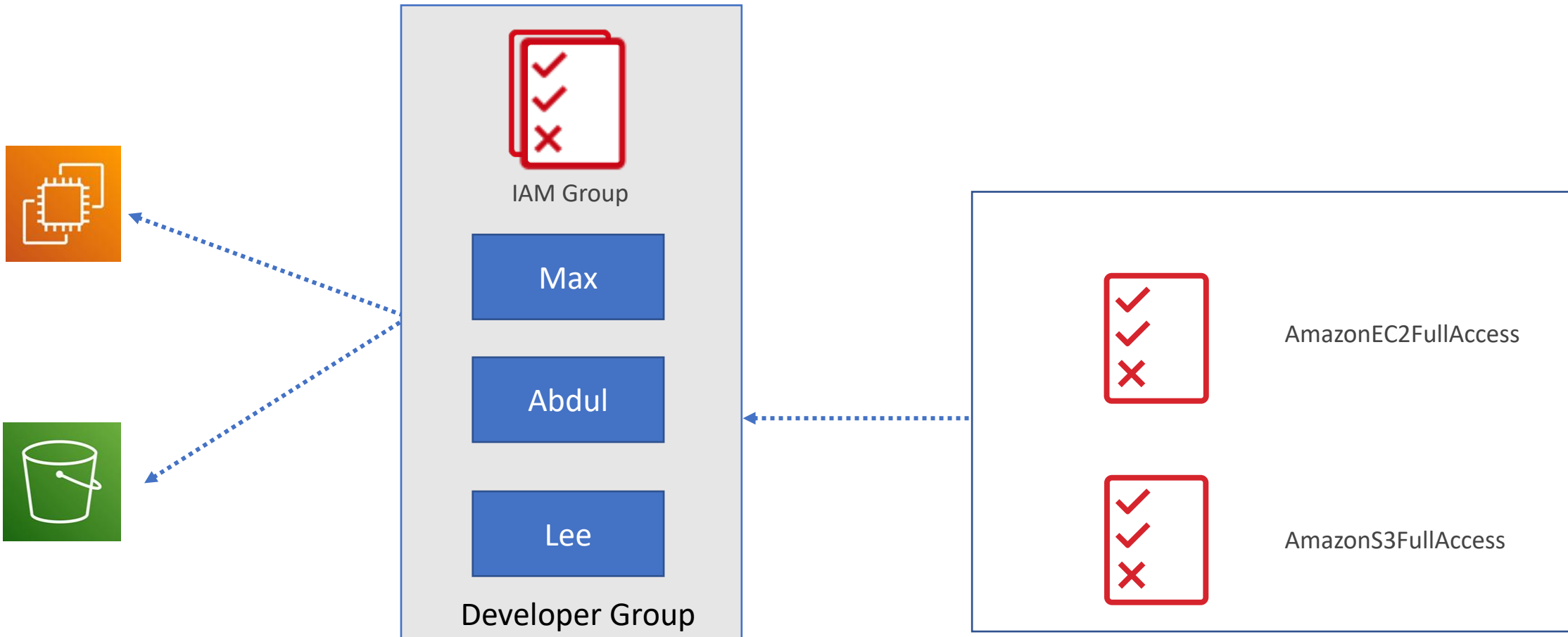
AdministratorAccess

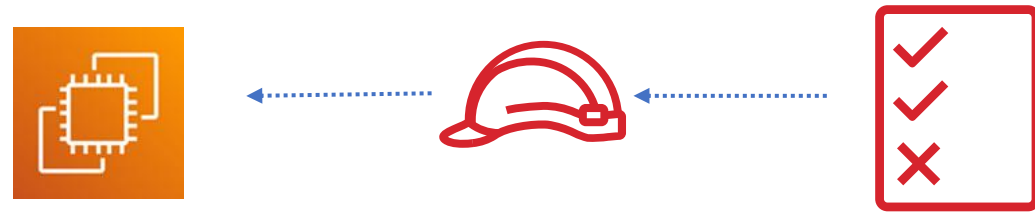
Filter policies ▾			Search	Showing 583 results	
		Policy name ▾	Type	Used as	
<input type="checkbox"/>	▶	 AdministratorAccess	Job function	Permissions policy (2)	
<input type="checkbox"/>	▶	 AlexaForBusinessDeviceSetup	AWS managed	None	
<input type="checkbox"/>	▶	 AlexaForBusinessFullAccess	AWS managed	None	
<input type="checkbox"/>	▶	 AlexaForBusinessGatewayExecution	AWS managed	None	
<input type="checkbox"/>	▶	 AlexaForBusinessLifesizeDelegatedAccessPolicy	AWS managed	None	
<input type="checkbox"/>	▶	 AlexaForBusinessPolyDelegatedAccessPolicy	AWS managed	None	
<input type="checkbox"/>	▶	 AlexaForBusinessReadOnlyAccess	AWS managed	None	
<input type="checkbox"/>	▶	 AmazonAPIGatewayAdministrator	AWS managed	None	

Job Function	Policy Name
Administrator	AdministratorAccess
Billing	Billing
Database Administrator	DatbaseAdministrator
Network Administrator	NetworkAdministrator
View-Only User	ViewOnlyAccess

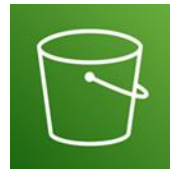
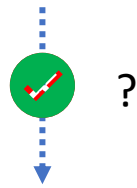


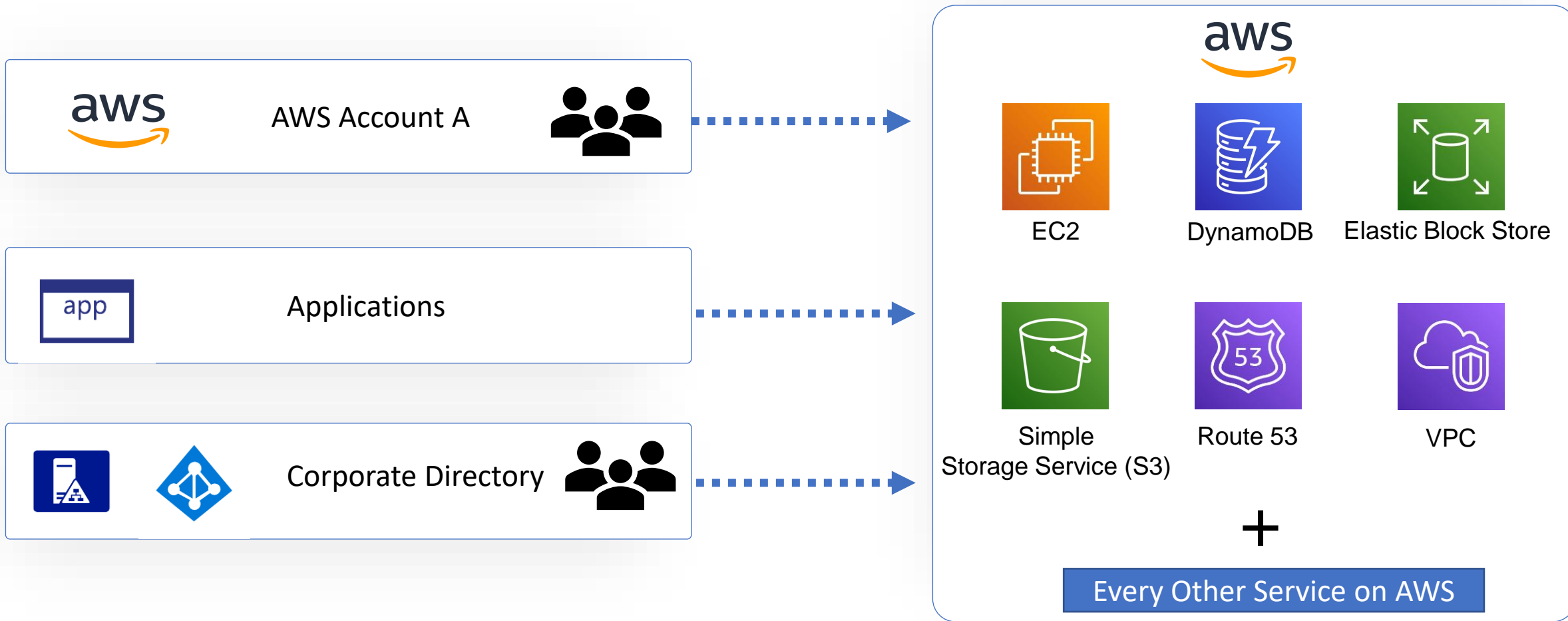






AmazonS3FullAccess







CreateEC2TagsPolicy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteTags",
        "ec2:CreateTags"
      ],
      "Resource": "*"
    }
  ]
}
```

Filter policies ▾

	Policy name ▾	Type	Used as	Description
<input type="radio"/>	CreateEC2TagsPolicy	Customer managed	None	Permission to create and delete EC2 Tags



Services ^

Resource Groups v



History

Console Home

VPC

EC2

S3

Find a service by name or feature (for example, EC2, S3 or VM, storage).



Compute

EC2
Lightsail
Lambda
Batch
Elastic Beanstalk
Serverless Application Repository
AWS Outposts
EC2 Image Builder



Storage

S3
EFS
FSx
S3 Glacier
Storage Gateway
AWS Backup



Blockchain

Amazon Managed Blockchain



Satellite

Ground Station



Quantum Technologies

Amazon Braket



Management & Governance

AWS Organizations
CloudWatch
AWS Auto Scaling
CloudFormation
CloudTrail
Config



Analytics

Athena
EMR
CloudSearch
Elasticsearch Service
Kinesis
QuickSight
Data Pipeline
AWS Data Exchange
AWS Glue
AWS Lake Formation
MSK



Security, Identity, & Compliance

IAM
Resource Access Manager
Cognito



Business Applications

Alexa for Business
Amazon Chime
WorkMail
Amazon Honeycode



End User Computing

WorkSpaces
AppStream 2.0
WorkDocs
WorkLink



Internet Of Things

IoT Core
FreeRTOS
IoT 1-Click
IoT Analytics

Identity and Access Management (IAM)

Dashboard

▼ Access management

Groups

Users

Roles



Welcome to Identity and Access Management

IAM users sign-in link:

<https://kodekloud.signin.aws.amazon.com/console>

IAM Resources

Users: 4

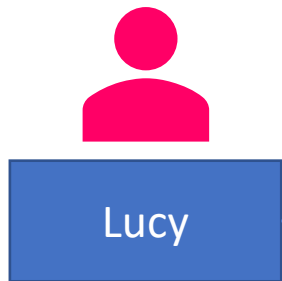
Groups: 2

Customer Managed Policies: 3

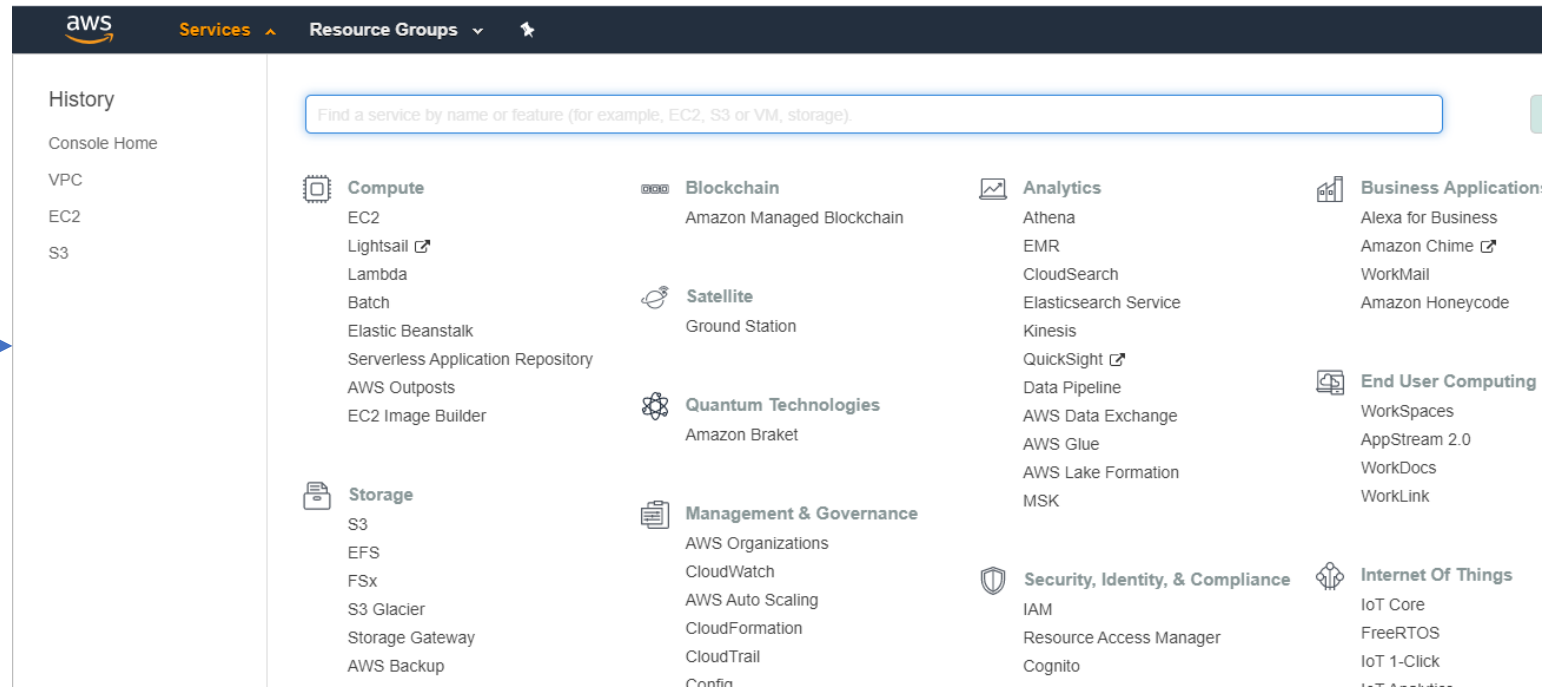


The background is a solid blue color. In the center, there are several concentric, rounded hexagonal shapes in a lighter shade of blue. Additionally, there are some geometric shapes in the corners: a light blue trapezoid in the top-left and a light blue parallelogram in the bottom-right.

Programmatic Access



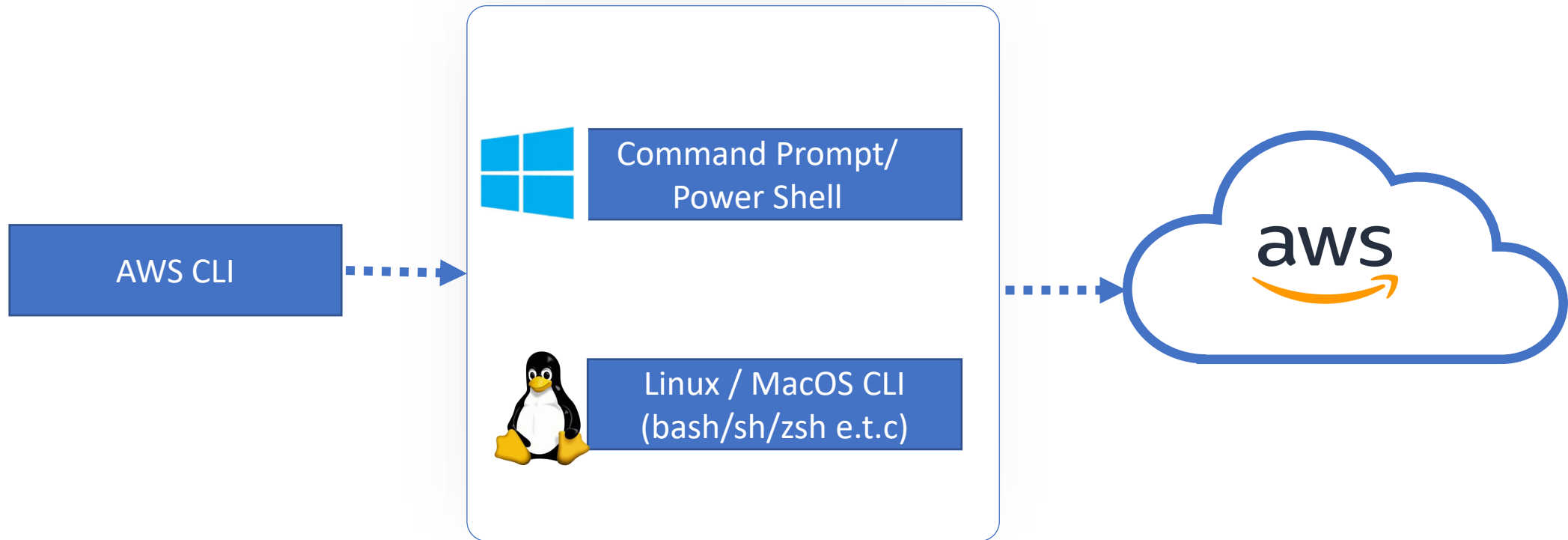
Username
Password



console.aws.com

```
$ aws s3api create-bucket --bucket my-bucket --region us-east-1
```

Access Key ID
Secret Access Key





```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip" -o "awscliv2.zip"
$ unzip awscliv2.zip
$ sudo ./aws/install

$ aws --version
aws-cli/2.0.47 Python/3.7.4 Linux/4.14.133-113.105.amzn2.x86_64 botocore/2.0.0
```



Download and Install <https://awscli.amazonaws.com/AWSCLIV2.msi>

```
C:\> aws --version
aws-cli/2.0.47 Python/3.7.4 Windows/10 botocore/2.0.0
```



Download and Install

<https://awscli.amazonaws.com/AWSCLIV2.pkg>

```
$ aws --version
aws-cli/2.0.47 Python/3.7.4 Darwin/18.7.0 botocore/2.0.0
```

```
$ aws configure
AWS Access Key ID [None]: AKIAI44QH8DHBEXAMPLE

AWS Secret Access Key [None]: je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY

Default region name [None]: us-west-2

Default output format [None]: json
```

```
$ cat .aws/config/config
[default]
region = us-west-2
output = text
```

```
$ cat .aws/config/credentials
[default]
aws_access_key_id = AKIAI44QH8DHBEXAMPLE
aws_secret_access_key = je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

```
$ aws <command> <subcommand> [options and parameters]
```

```
$ aws iam create-user --user-name lucy
{
  "User": {
    "UserName": "lucy",
    "Tags": [],
    "CreateDate": "2020-09-15T23:40:11.168Z",
    "UserId": "h9r2sc5br8ss7uzhs2qm",
    "Path": "/",
    "Arn": "arn:aws:iam::000000000000:user/lucy"
  }
}
```

Command	Value
command	iam
subcommand	create-user
option	--user-name
parameter	lucy

```
$ aws help
```

```
AWS()
```

```
NAME
```

```
aws -
```

```
DESCRIPTION
```

```
The AWS Command Line Interface is a unified tool to  
manage your AWS  
services.
```

```
SYNOPSIS
```

```
aws [options] <command> <subcommand> [parameters]
```

```
Use aws command help for information on a specific  
command. Use aws  
help topics to view a list of available help topics. The  
synopsis for  
each command shows its parameters and their usage.  
Optional parameters  
are shown in square brackets.
```

```
.  
.
```

```
[Output Truncated]
```

```
$ aws iam help
```

```
IAM()
```

```
NAME
```

```
    iam -
```

```
DESCRIPTION
```

```
    AWS Identity and Access Management (IAM) is a web service for securely
```

```
    controlling access to AWS services. With IAM, you can centrally manage
```

```
    users, security credentials such as access keys, and permissions that
```

```
    control which AWS resources users and applications can access. For more
```

```
    information about IAM, see AWS Identity and Access Management (IAM) and
```

```
    the AWS Identity and Access Management User Guide .
```

```
AVAILABLE COMMANDS
```

```
    o add-client-id-to-open-id-connect-provider
```

```
    o add-role-to-instance-profile
```

```
    .
```

```
    .
```

```
[Output Truncated]
```

```
$ aws <command> help
```

```
$ aws iam create-user help
```

NAME

create-user -

DESCRIPTION

Creates a new IAM user for your AWS account.

The number and size of IAM resources in an AWS account are limited. For more information, see IAM and STS Quotas in the IAM User Guide .

See also: AWS API Documentation

See 'aws help' for descriptions of global parameters.

SYNOPSIS

```
    create-user  
    [--path <value>]  
    --user-name <value>  
    [--permissions-boundary <value>]  
    [--tags <value>]  
    [--cli-input-json <value>]  
    [--generate-cli-skeleton <value>]
```

```
$ aws <command> <subcommand> help
```




IAM with Terraform

> GuardDuty

▼ IAM

▼ Resources

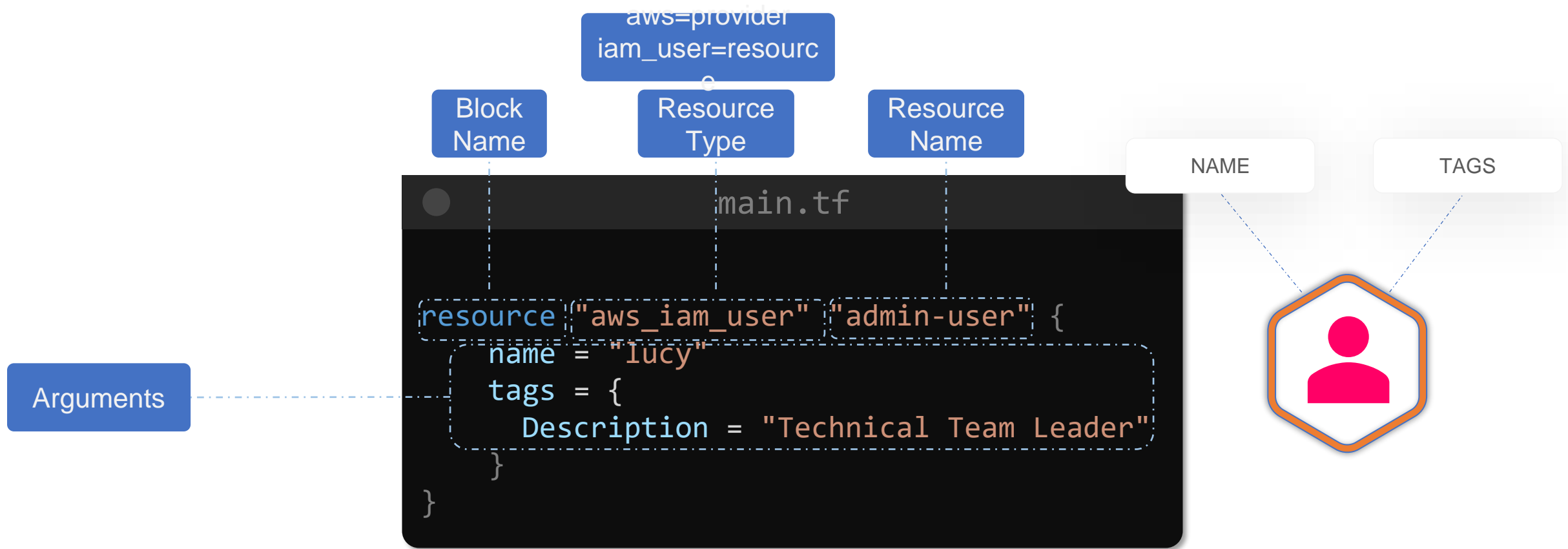
- aws_iam_access_key
- aws_iam_account_alias
- aws_iam_account_password_policy
- aws_iam_group
- aws_iam_role
- aws_iam_role_policy
- aws_iam_role_policy_attachment
- aws_iam_saml_provider
- aws_iam_server_certificate
- aws_iam_service_linked_role
- **aws_iam_user**
- aws_iam_user_group_membership
- aws_iam_user_login_profile
- aws_iam_user_policy

```
}  
EOF  
}
```

Argument Reference

The following arguments are supported:

- `name` - (Required) The user's name. The name must consist of upper and lowercase alphanumeric characters with no spaces. You can also include any of the following characters: `=",.@-_.`. User names are not distinguished by case. For example, you cannot create users named both "TESTUSER" and "testuser".
- `path` - (Optional, default `"/"`) Path in which to create the user.
- `permissions_boundary` - (Optional) The ARN of the policy that is used to set the permissions boundary for the user.
- `force_destroy` - (Optional, default `false`) When destroying this user, destroy even if it has non-Terraform-managed IAM access keys, login profile or MFA devices. Without `force_destroy` a user with non-Terraform-managed access keys and login profile will fail to be destroyed.
- `tags` - Key-value map of tags for the IAM user



main.tf

```
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Technical Team Leader"  
  }  
}
```



>_

```
$ terraform init
```

Initializing the backend...

Initializing provider plugins...

- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.6.0...
- Installed hashicorp/aws v3.6.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration, so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking changes, we recommend adding version constraints in a `required_providers` block in your configuration, with the constraint strings suggested below.

```
* hashicorp/aws: version = "~> 3.6.0"
```

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for

main.tf

```
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Technical Team Leader"  
  }  
}
```



>_

\$ terraform plan

provider.aws.region

The region where AWS operations will take place. Examples are us-east-1, us-west-2, etc.

Enter a value: **us-west-1**

Refreshing Terraform state in-memory prior to plan...

The refreshed state will be used to calculate this plan, but will not be persisted to local or remote state storage.

Error: error configuring Terraform AWS Provider: no valid credential sources for Terraform AWS Provider found.

Please see

<https://registry.terraform.io/providers/hashicorp/aws>
for more information about providing credentials.

Error: NoCredentialProviders: no valid providers in chain.
Deprecated.

For verbose messaging see
aws.Config.CredentialsChainVerboseErrors

main.tf

```
provider "aws" {  
    region = "us-west-2"  
    access_key = "AKIAI44QH8DHBEXAMPLE"  
    secret_key = "je7MtGbClwBF/2tk/h3yCo8n..."  
}  
  
resource "aws_iam_user" "admin-user" {  
    name = "lucy"  
    tags = {  
        Description = "Technical Team Leader"  
    }  
}
```



>_

\$ terraform plan

.
.
+ create

Terraform will perform the following actions:

```
# aws_iam_user.admin-user will be created  
+ resource "aws_iam_user" "admin-user" {  
    + arn              = (known after apply)  
    + force_destroy    = false  
    + id               = (known after apply)  
    + name             = "Lucy"  
    + path             = "/"  
    + tags             = {  
        + "Description" = "Technical Team Lead"  
    }  
    + unique_id        = (known after apply)  
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't specify an "-out" parameter to save this

main.tf

```
provider "aws" {  
    region = "us-west-2"  
    access_key = "AKIAI44QH8DHBEXAMPLE"  
    secret_key = "je7MtGbClwBF/2tk/h3yCo8n..."  
}  
  
resource "aws_iam_user" "admin-user" {  
    name = "lucy"  
    tags = {  
        Description = "Technical Team Leader"  
    }  
}
```



>_

\$ terraform apply

```
# aws_iam_user.admin-user will be created  
+ resource "aws_iam_user" "admin-user" {  
    + arn              = (known after apply)  
    + force_destroy    = false  
    + id               = (known after apply)  
    + name              = "Lucy"  
    + path              = "/"  
    + tags              = {  
        + "Description" = "Technical Team Lead"  
    }  
    + unique_id         = (known after apply)  
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

```
aws_iam_user.admin-user: Creating...  
aws_iam_user.admin-user: Creation complete after 1s  
[id=Lucy]
```

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

main.tf

```
provider "aws" {  
    region = "us-west-2"  
    access_key = "AKIAI44QH8DHBEXAMPLE"  
    secret_key = "je7MtGbClwBF/2tk/h3yCo8n..."  
}  
  
resource "aws_iam_user" "admin-user" {  
    name = "lucy"  
    tags = {  
        Description = "Technical Team Leader"  
    }  
}
```

.aws/config/credentials

```
[default]  
aws_access_key_id =  
aws_secret_access_key =
```

main.tf

```
provider "aws" {  
    region = "us-west-2"  
}  
  
resource "aws_iam_user" "admin-user" {  
    name = "lucy"  
    tags = {  
        Description = "Technical Team Leader"  
    }  
}
```

.aws/config/credentials

```
[default]  
aws_access_key_id = AKIAI44QH8DHBEXAMPLE  
aws_secret_access_key = je7MtGbClwBF/2tk/h3yCo8n...
```

> _

```
$ export AWS_ACCESS_KEY_ID=  
$ export AWS_SECRET_ACCESS_KEY_ID=
```

main.tf

```
provider "aws" {  
    region = "us-west-2"  
}  
  
resource "aws_iam_user" "admin-user" {  
    name = "lucy"  
    tags = {  
        Description = "Technical Team Leader"  
    }  
}
```

.aws/config/credentials

```
[default]  
aws_access_key_id =  
aws_secret_access_key =
```

> _

```
$ export AWS_ACCESS_KEY_ID=AKIAI44QH8DHBEXAMPLE  
$ export AWS_SECRET_ACCESS_KEY_ID=je7MtGbClwBF/2tk/h3yCo8n...  
$ export AWS_REGION=us-west-2
```



{K}ODE{K}LOUD

IAM policies with Terraform

main.tf

```
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Technical Team Leader"  
  }  
}
```



```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "*",  
      "Resource": "*"  
    }  
  ]  
}
```

AdministratorAccess
Policy

Argument Reference

The following arguments are supported:

- `description` - (Optional, Forces new resource) Description of the IAM policy.
- `name` - (Optional, Forces new resource) The name of the policy. If omitted, Terraform will assign a random, unique name.
- `name_prefix` - (Optional, Forces new resource) Creates a unique name beginning with the specified prefix. Conflicts with `name`.
- `path` - (Optional, default "/") Path in which to create the policy. See [IAM Identifiers](#) for more information.
- `policy` - (Required) The policy document. This is a JSON formatted string. For more information about building AWS IAM policy documents with Terraform, see the [AWS IAM Policy Document Guide](#)

main.tf

```
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Technical Team Leader"  
  }  
}  
  
resource "aws_iam_policy" "adminUser" {  
  name      = "AdminUsers"  
  policy    = ?  
}
```



IAM Policy

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "*",  
      "Resource": "*"  
    }  
  ]  
}
```

AdministratorAccess
Policy

```
[COMMAND] <<DELIMITER  
Line1  
Line2  
Line3  
DELIMITER
```

Heredoc Syntax

main.tf

```
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Technical Team Leader"  
  }  
}  
  
resource "aws_iam_policy" "adminUser" {  
  name      = "AdminUsers"  
  policy    = <<EOF  
  
  EOF  
}
```



IAM Policy

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "*",  
      "Resource": "*"  
    }  
  ]  
}
```

AdministratorAccess
Policy

```
[COMMAND] <<DELIMITER  
Line1  
Line2  
Line3  
DELIMITER
```

Heredoc Syntax

main.tf

```
resource "aws_iam_user" "admin-user" {  
  name = "lucy"  
  tags = {  
    Description = "Technical Team Leader"  
  }  
}  
  
resource "aws_iam_policy" "adminUser" {  
  name = "AdminUsers"  
  policy = <<EOF  
  {  
    "Version": "2012-10-17",  
    "Statement": [  
      {  
        "Effect": "Allow",  
        "Action": "*",  
        "Resource": "*"   
      }  
    ]  
  }  
  EOF  
}  
  
resource "aws_iam_user_policy_attachment" "lucy-admin-access" {  
  user = aws_iam_user.admin-user.name  
  policy_arn = aws_iam_policy.adminUser.arn  
}
```

AdministratorAccess
Policy

```
[COMMAND] <<DELIMITER  
Line1  
Line2  
Line3  
DELIMITER
```

Heredoc Syntax



??



IAM Policy

main.tf

```
resource "aws_iam_user" "admin-user" {
  name = "lucy"
  tags = {
    Description = "Technical Team Leader"
  }
}

resource "aws_iam_policy" "adminUser" {
  name = "AdminUsers"
  policy = <<EOF
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": "*",
        "Resource": "*"
      }
    ]
  }
  EOF
}

resource "aws_iam_user_policy_attachment" "lucy-admin-access" {
  user = aws_iam_user.admin-user.name
  policy_arn = aws_iam_policy.adminUser.arn
}
```

>_

\$ terraform apply

```
# aws_iam_policy.adminUser will be created
+ resource "aws_iam_policy" "adminUser" {
  + arn      = (known after apply)
  + id       = (known after apply)
  + name     = "AdminUsers"
  + path     = "/"
  + policy   = jsonencode(
    {
      + Statement = [
        + {
          + Action    = "*"
          + Effect    = "Allow"
          + Resource   = "*"
        },
      ]
      + Version     = "2012-10-17"
    }
  )
}

.[Output Truncated]
aws_iam_user.lucy: Creating...
aws_iam_policy.adminUser: Creating...
aws_iam_user.lucy: Creation complete after 0s [id=lucy]
aws_iam_policy.adminUser: Creation complete after 0s
[id=arn:aws:iam::000000000000:policy/AdminUsers]
aws_iam_user_policy_attachment.lucy-admin-access: Creating...
aws_iam_user_policy_attachment.lucy-admin-access: Creation complete
after 0s [id=lucy-20200919034158686100000001]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

main.tf

```
resource "aws_iam_user" "admin-user" {
  name = "lucy"
  tags = {
    Description = "Technical Team Leader"
  }
}

resource "aws_iam_policy" "adminUser" {
  name     = "AdminUsers"
  policy = <<EOF
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": "*",
        "Resource": "*"
      }
    ]
  }
  EOF
}

resource "aws_iam_user_policy_attachment" "lucy-admin-access" {
  user = aws_iam_user.admin-user.name

  policy_arn = aws_iam_policy.adminUser.arn
}
```

admin-policy.json

main.tf

```
resource "aws_iam_user" "admin-user" {
  name = "lucy"
  tags = {
    Description = "Technical Team Leader"
  }
}

resource "aws_iam_policy" "adminUser" {
  name     = "AdminUsers"
  policy   = file("admin-policy.json")

  EOF
}

resource "aws_iam_user_policy_attachment" "lucy-admin-access" {

  user = aws_iam_user.admin-user.name

  policy_arn = aws_iam_policy.adminUser.arn
}
```

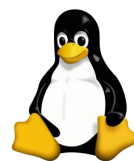
admin-policy.json

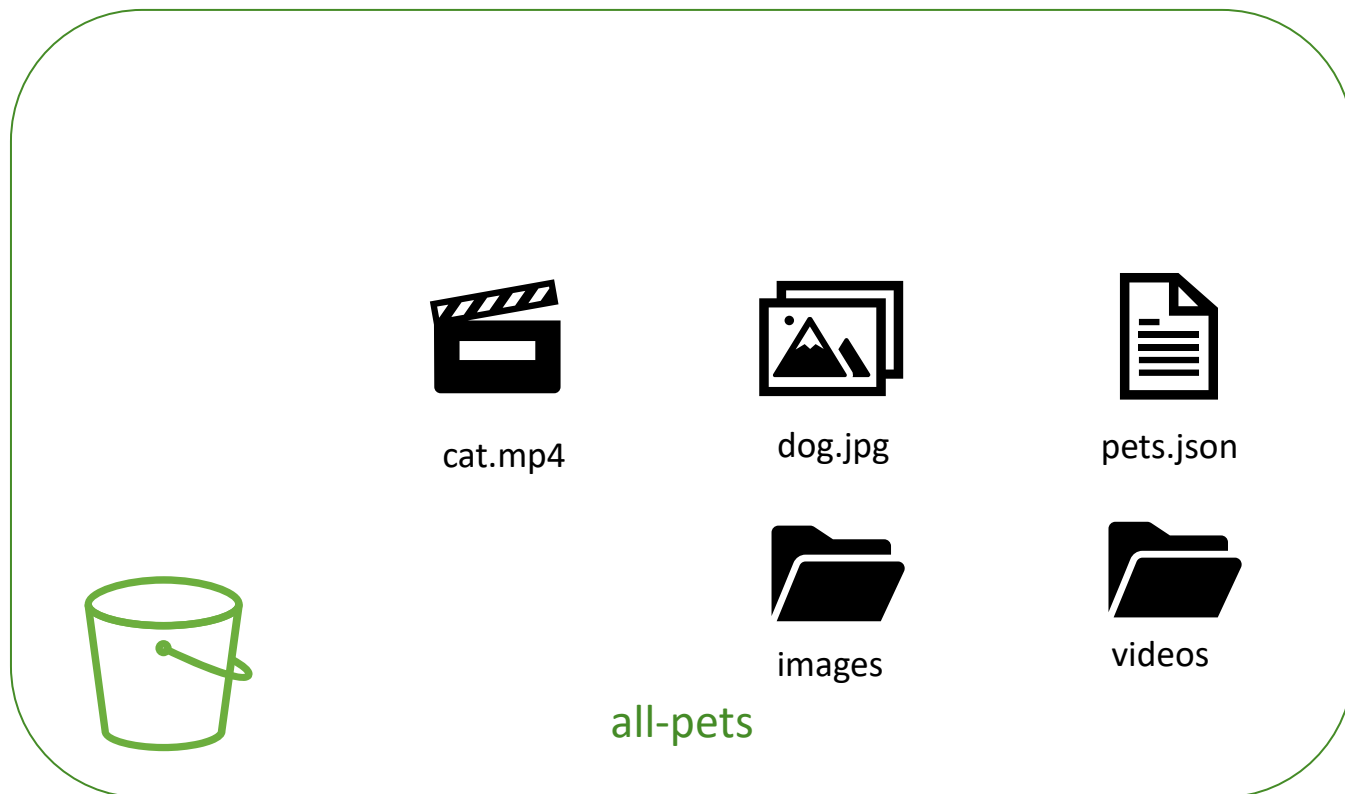
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```



The image features the AWS S3 logo, which consists of a large, light blue hexagon with rounded corners. Inside this hexagon are several concentric, slightly offset hexagons of the same color, creating a tunnel-like effect. The text "AWS S3" is centered within the innermost hexagon in a white, bold, sans-serif font.

AWS S3





Object #	Name
1	pets.json
2	dog.jpg
3	cat.mp4
4	pictures/cat.jpg
5	videos/dog.mp4

Unique Bucket Name

DNS Compliant Name

Files size between 0 to
5TB

The screenshot shows the 'Create bucket' wizard in the AWS Management Console. The title bar is blue with a close button (X) on the right. Below the title bar is a progress bar with four steps: 1. Name and region (active), 2. Configure options, 3. Set permissions, and 4. Review. The main content area is dark blue and contains the following fields:

- Name and region**
 - Bucket name**: A text input field with a blue information icon. The placeholder text is 'Enter DNS-compliant bucket name'.
 - Region**: A dropdown menu showing 'US West (Oregon)' with a blue downward arrow.
- Copy settings from an existing bucket**: A section with a text input field. The placeholder text is 'Select bucket (optional)' followed by '20 Buckets' and a blue downward arrow.

At the bottom of the form, there are three buttons: 'Create' (white with blue text), 'Cancel' (blue with white text), and 'Next' (grey with white text).

`https://<bucket_name>.<region>.amazonaws.com`

`https://all-pets.us-west-1.amazonaws.com`

Object #	Name	Address
1	pets.json	https://all-pets.us-west-1.amazonaws.com/pets.json
2	dog.jpg	https://all-pets.us-west-1.amazonaws.com/dog.jpg
3	cat.mp4	https://all-pets.us-west-1.amazonaws.com/cat.mp4
4	pictures/cat.jpg	https://all-pets.us-west-1.amazonaws.com/pictures/cat.jpg
5	videos/dog.mp4	https://all-pets.us-west-1.amazonaws.com/videos/dog.mp4



dog.jpg

Key = dog.jpg

Value = Data

Object data

Owner = Lucy

Size = 5MB

Last Modified = Jan 26, 2020 12:55:21 AM GMT-0500

Metadata

all-pets

Access Control Lists



dog.jpg



Bucket Policies

all-pets





Lucy



Access Control Lists



dog.jpg



Bucket Policies



all-pets

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::all-pets/*",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456123457:user/Lucy"
        ]
      }
    }
  ]
}
```

read-objects.json



{K}ODE{K}LOUD

S3 with Terraform

main.tf

```
resource "aws_s3_bucket" "finance" {  
    bucket = "finanace-21092020"  
    tags   = {  
        Description = "Finance and Payroll"  
    }  
}
```

>_

```
$ terraform apply
```

Terraform will perform the following actions:

```
# aws_s3_bucket.finance will be created  
+ resource "aws_s3_bucket" "finance" {  
    + acceleration_status = (known after apply)  
    + acl                  = "private"  
    + arn                  = (known after apply)  
    + bucket                = "finanace-21092020"
```

```
.  
.
```

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_s3_bucket.finance: Creating...

aws_s3_bucket.finance: Creation complete after 0s
[id=finanace-21092020]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

main.tf

```
resource "aws_s3_bucket" "finance" {  
  bucket = "finanace-21092020"  
  tags = {  
    Description = "Finance and Payroll"  
  }  
}  
  
resource "aws_s3_bucket_object" "finance-2020" {  
  content = "/root/finance/finance-2020.doc"  
  key     = "finance-2020.doc"  
  bucket = aws_s3_bucket.finance.id  
}
```

>_

\$ terraform apply

.
.

Terraform will perform the following actions:

```
# aws_s3_bucket_object.finance-2020 will be created  
+ resource "aws_s3_bucket_object" "finance-2020" {  
  + acl                = "private"  
  + bucket              = "finanace-21092020"  
  + content             = "/root/finance/finance-  
2020.doc"  
+ force_destroy        = false  
  + id                  = (known after apply)  
  + key                 = "finance/finance-  
2020.doc"
```

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```
aws_s3_bucket_object.finance-2020: Creating...  
aws_s3_bucket_object.finance-2020: Creation complete  
after 0s [id=finance/finance-2020.doc]
```

main.tf

```
resource "aws_s3_bucket" "finance" {  
  bucket = "finanace-21092020"  
  tags = {  
    Description = "Finance and Payroll"  
  }  
}  
  
resource "aws_s3_bucket_object" "finance-2020" {  
  content = "/root/finance/finance-2020.doc"  
  key     = "finance-2020.doc"  
  bucket = aws_s3_bucket.finance.id  
}  
  
data "aws_iam_group" "finance-data" {  
  group_name = "finance-analysts"  
}
```

AWS



finance-21092020

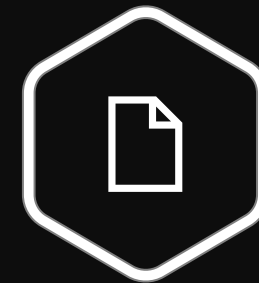


finance-2020.doc



finance-analysts

terraform.tfstate



finance-data

```

    content = "/root/finance/finance-2020.doc"
    key      = "finance-2020.doc"
    bucket = aws_s3_bucket.finance.id
}

data "aws_iam_group" "finance-data" {
    group_name = "finance-analysts"
}

resource "aws_s3_bucket_policy" "finance-policy" {
    bucket = aws_s3_bucket.finance.id
    policy = <<EOF

```

```

EOF
}

```

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::<<bucket-name>>/*",
      "Principal": {
        "AWS": [
          "<< arn >>"
        ]
      }
    }
  ]
}

```

read-objects.json

```

content = "/root/finance/finance-2020.doc"
key      = "finance-2020.doc"
bucket = aws_s3_bucket.finance.id
}

data "aws_iam_group" "finance-data" {
  group_name = "finance-analysts"
}

resource "aws_s3_bucket_policy" "finance-policy" {
  bucket = aws_s3_bucket.finance.id
  policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::${aws_s3_bucket.finance.id}/*",
      "Principal": {
        "AWS": [
          "${data.aws_iam_group.finance-data.arn}"
        ]
      }
    }
  ]
}
]
}
EOF
}

```

> _

\$ terraform apply

.
.

Terraform will perform the following actions:

```
# aws_s3_bucket_object.finance-2020 will be created
+ resource "aws_s3_bucket_object" "finance-2020" {
  + acl                = "private"
  + bucket              = "finanace-21092020"
  + content             = "/root/finance/finance-2020.doc"
+ force_destroy        = false
  + id                 = (known after apply)
  + key                 = "finance/finance-2020.doc"
```

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.

Only 'yes' will be accepted to approve.

Enter a value: yes

aws_s3_bucket_object.finance-2020: Creating...

aws_s3_bucket_object.finance-2020: Creation complete after 0s

[id=finance/finance-2020.doc]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.



Introduction to DynamoDB



DynamoDB

Highly Scalable

Fully Managed by AWS

NoSQL Database

Single-Digit MilliSecond Latency

Data Replicated across Regions

Manufacturer	Model
Toyota	Corolla
Honda	Civic
Dodge	Journey
Ford	F150

	Manufacturer	Model	Year	VIN
Item 1	Toyota	Corolla	2004	4Y1SL65848Z411439
Item 2	Honda	Civic	2017	DY1SL65848Z411432
Item 3	Dodge	Journey	2014	SD1SL65848Z411443
Item 4	Ford	F150	2020	DH1SL65848Z41100

Manufacturer	Model	Year	VIN
Toyota	Corolla	2004	4Y1SL65848Z411439
Honda	Civic	2017	DY1SL65848Z411432
Dodge	Journey	2014	SD1SL65848Z411443
Ford	F150	2020	DH1SL65848Z41100

cars

```
{
  "Manufacturer": "Toyota",
  "Make": "Corolla",
  "Year": 2004,
  "VIN" : "4Y1SL65848Z411439"
}
```

```
{
  "Manufacturer": "Honda",
  "Make": "Civic",
  "Year": 2017,
  "VIN" : "DY1SL65848Z411432"
}
```

```
{
  "Manufacturer": "Dodge",
  "Make": "Journey",
  "Year": 2014,
  "VIN" : "SD1SL65848Z411443"
}
```

```
{
  "Manufacturer": "Ford",
  "Make": "F150",
  "Year": 2020,
  "VIN" : "DH1SL65848Z41100"
}
```

PRIMARY KEY

Manufacturer	Model	Year	VIN
Toyota	Corolla	2004	4Y1SL65848Z411439
Honda	Civic	2017	DY1SL65848Z411432
Dodge	Journey	2014	SD1SL65848Z411443
Ford	F150	2020	DH1SL65848Z41100

```
{
  "Manufacturer": "Honda",
  "Make": "Civic",
  "Year": 2017,
  "VIN" : "DY1SL65848Z411432"
}
```

```
{
  "Manufacturer": "Dodge",
  "Make": "Journey",
  "Year": 2014,
  "VIN" : "SD1SL65848Z411443"
}
```

```
{
  "Manufacturer": "Ford",
  "Make": "F150",
  "Year": 2020,
  "VIN" : "DH1SL65848Z41100"
}
```

```
{
  "Manufacturer": "Jaguar",
  "Make": "",
  "Year": "",
  "VIN" : "LB1SL65848Z41123"
}
```



DynamoDB with Terraform

main.tf

```
resource "aws_dynamodb_table" "cars" {  
  name      = "cars"  
  hash_key  = "VIN"  
  billing_mode = "PAY_PER_REQUEST"  
  attribute {  
    name = "VIN"  
    type = "S"  
  }  
}
```

- `billing_mode` - (Optional) Controls how you are charged for read and write throughput and how you manage capacity. The valid values are `PROVISIONED` and `PAY_PER_REQUEST`. Defaults to `PROVISIONED`.
- `write_capacity` - (Optional) The number of write units for this table. If the `billing_mode` is `PROVISIONED`, this field is required.
- `read_capacity` - (Optional) The number of read units for this table. If the `billing_mode` is `PROVISIONED`, this field is required.

main.tf

```
resource "aws_dynamodb_table" "cars" {  
  name      = "cars"  
  hash_key  = "VIN"  
  billing_mode = "PAY_PER_REQUEST"  
  attribute {  
    name = "VIN"  
    type = "S"  
  }  
}
```

>_

```
$ terraform apply
```

```
+ create
```

Terraform will perform the following actions:

```
# aws_dynamodb_table.cars will be created  
+ resource "aws_dynamodb_table" "cars" {  
  + arn              = (known after apply)  
  + billing_mode     = "PAY_PER_REQUEST"  
  + hash_key         = "VIN"  
  + id              = (known after apply)  
  + name            = "cars"  
  + stream_arn       = (known after apply)  
  + stream_label     = (known after apply)  
  + stream_view_type = (known after apply)  
  
  + attribute {  
    + name = "VIN"  
    + type = "S"  
  }  
  
  + point_in_time_recovery {  
    + enabled = (known after apply)  
  }  
}
```

```
.  
.
```

```
aws_dynamodb_table.cars: Creating...
```

```
aws_dynamodb_table.cars: Creation complete after 0s [id=cars]
```


main.tf

```
resource "aws_dynamodb_table" "cars" {
  name      = "cars"
  hash_key  = "VIN"
  billing_mode = "PAY_PER_REQUEST"
  attribute {
    name = "VIN"
    type = "S"
  }
}

resource "aws_dynamodb_table_item" "car-items" {
  table_name = aws_dynamodb_table.cars.name
  hash_key   = aws_dynamodb_table.cars.hash_key
  item       = <<EOF

EOF
}
```

cars

```
{
  "Manufacturer": "Toyota",
  "Make": "Corolla",
  "Year": 2004,
  "VIN" : "4Y1SL65848Z411439"
}
```

```
{
  "Manufacturer": "Honda",
  "Make": "Civic",
  "Year": 2017,
  "VIN" : "DY1SL65848Z411432"
}
```

```
{
  "Manufacturer": "Dodge",
  "Make": "Journey",
  "Year": 2014,
  "VIN" : "SD1SL65848Z411443"
}
```

```
{
  "Manufacturer": "Ford",
  "Make": "F150",
  "Year": 2020,
  "VIN" : "DH1SL65848Z41100"
}
```

main.tf

```
resource "aws_dynamodb_table" "cars" {
  name      = "cars"
  hash_key  = "VIN"
  billing_mode = "PAY_PER_REQUEST"
  attribute {
    name = "VIN"
    type = "S"
  }
}

resource "aws_dynamodb_table_item" "car-items" {
  table_name = aws_dynamodb_table.cars.name
  hash_key   = aws_dynamodb_table.cars.hash_key
  item       = <<EOF

  {
    "Manufacturer": {"S": "Toyota"},
    "Make": {"S": "Corolla"},
    "Year": {"N": "2004"},
    "VIN" : {"S": "4Y1SL65848Z411439"},
  }
EOF
}
```

>_

\$ terraform apply

```
# aws_dynamodb_table_item.car-items will be created
+ resource "aws_dynamodb_table_item" "car-items" {
  + hash_key   = "VIN"
  + id         = (known after apply)
  + item       = jsonencode(
    {
      + Manufacturer = {
        + S = "Toyota"
      }
      + Model         = {
        + S = "Corolla"
      }
      + VIN           = {
        + S = "4Y1SL65848Z411439"
      }
      + Year          = {
        + N = "2004"
      }
    }
  )
  + table_name = "cars"
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

.

.

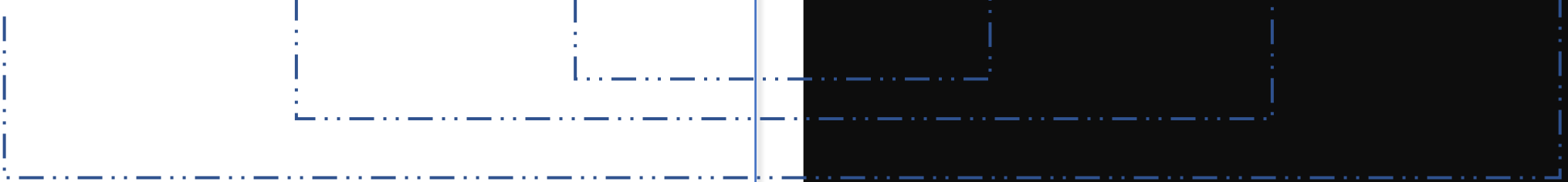
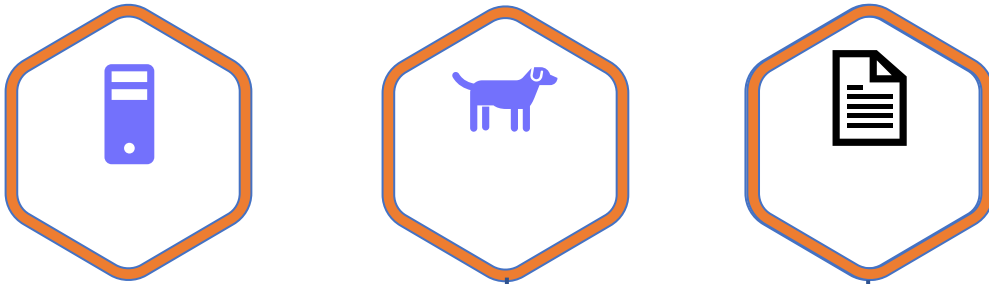
aws_dynamodb_table_item.car-items: Creating...



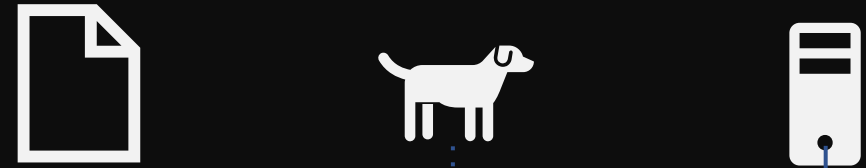
{K}ODE{K}LOUD

Remote State

Real World Infrastructure



terraform.tfstate



Mapping Configuration to Real World

Tracking Metadata

Performance

Collaboration

```
> _
```

```
$ ls
```

```
main.tf variables.tf terraform.tfstate
```

Version Control



Remote State Backends

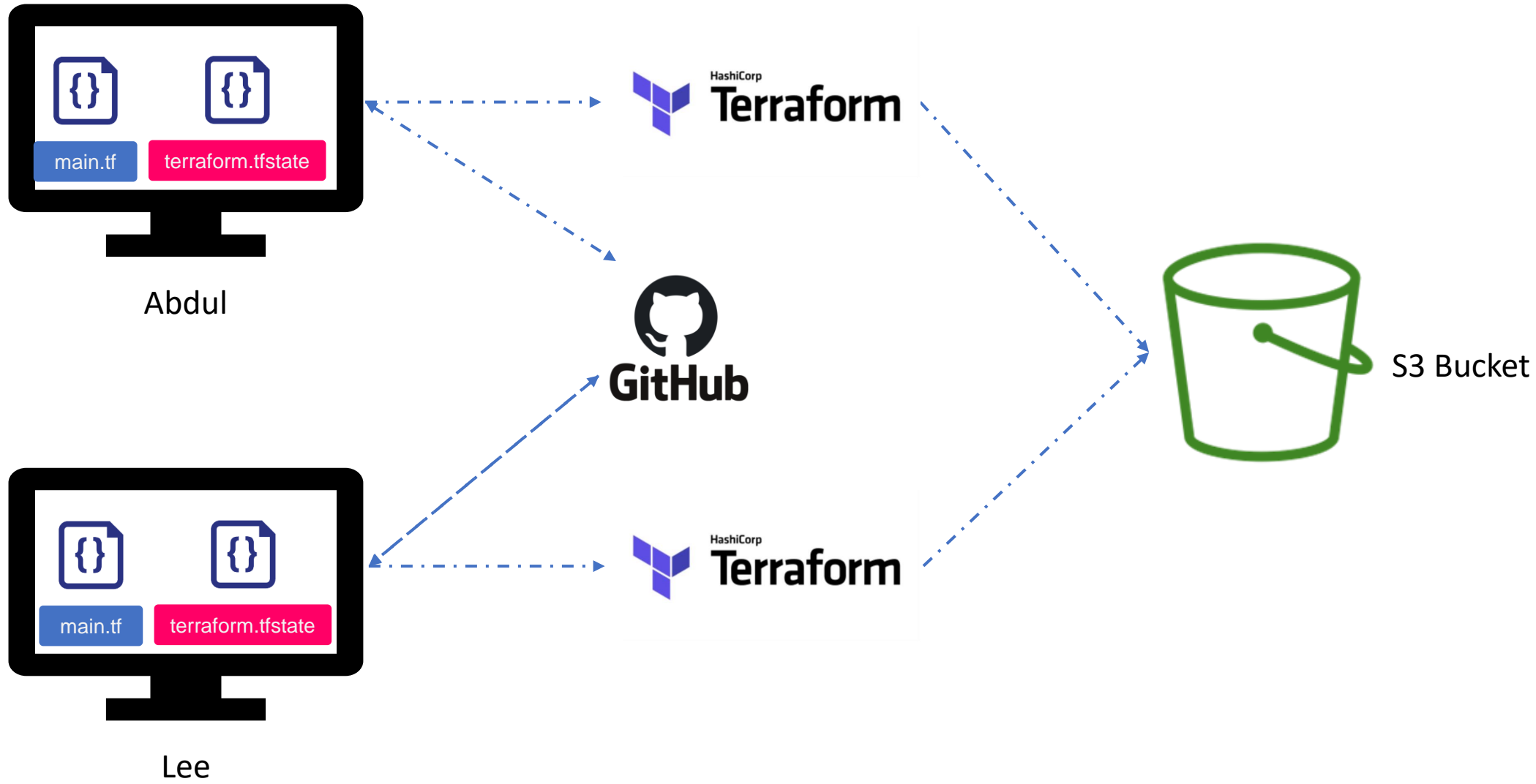


main.tf

```
resource "local_file" "pet" {
  filename = "/root/pet.txt"
  content  = "My favorite pet is Mr.Whiskers!"
}
resource "random_pet" "my-pet" {
  length = 1
}
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```

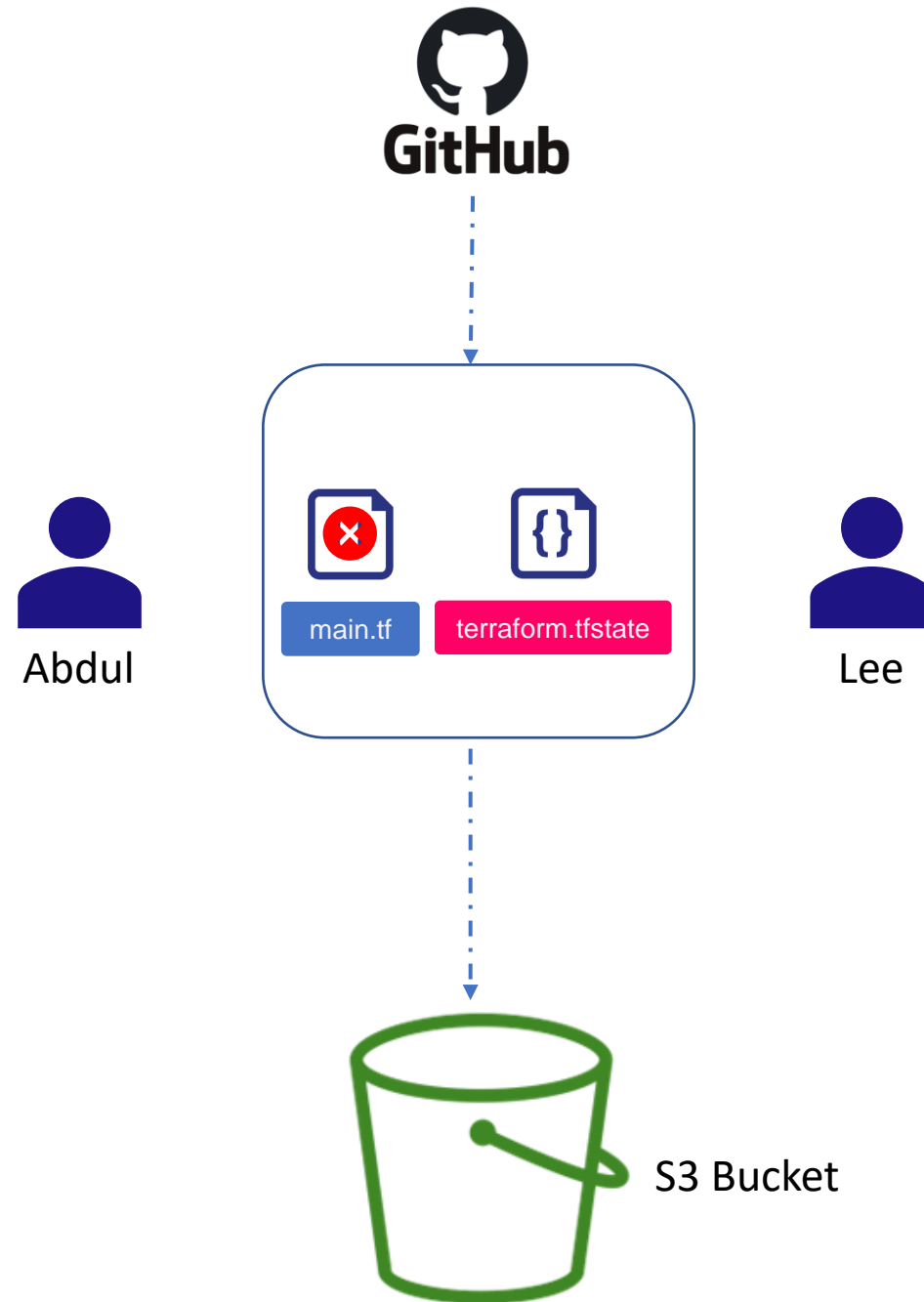
terraform.tfstate

```
{
  "mode": "managed",
  "type": "aws_instance",
  "name": "dev-ec2",
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
  "instances": [
    {
      "schema_version": 1,
      "attributes": {
        "ami": "ami-0a634ae95e11c6f91",
        .
        .
        .
        "primary_network_interface_id": "eni-0ccd57b1597e633e0",
        "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
        "private_ip": "172.31.7.21",
        "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
        "public_ip": "54.71.34.19",
        "root_block_device": [
          {
            "delete_on_termination": true,
            "device_name": "/dev/sda1",
            "encrypted": false,
            "iops": 100,
            "kms_key_id": "",
            "volume_id": "vol-070720a3636979c22",
            "volume_size": 8,
            "volume_type": "gp2"
          }
        ]
      }
    }
  ]
}
```



terraform.tfstate

```
{
  "mode": "managed",
  "type": "aws_instance",
  "name": "dev-ec2",
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
  "instances": [
    {
      "schema_version": 1,
      "attributes": {
        "ami": "ami-0a634ae95e11c6f91",
        .
        .
        .
        "primary_network_interface_id": "eni-0ccd57b1597e633e0",
        "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
        "private_ip": "172.31.7.21",
        "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
        "public_ip": "54.71.34.19",
        "root_block_device": [
          {
            "delete_on_termination": true,
            "device_name": "/dev/sda1",
            "encrypted": false,
            "iops": 100,
            "kms_key_id": "",
            "volume_id": "vol-070720a3636979c22",
            "volume_size": 8,
            "volume_type": "gp2"
          }
        ]
      }
    }
  ],
}
```



>_ Terminal 1

\$ terraform apply

```
.
.
."
+ server_side_encryption = (known after apply)
+ storage_class          = (known after apply)
+ version_id             = (known after apply)
}
```

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```
aws_s3_bucket_object.finance-2020: Creating...
aws_s3_bucket.finance: Creating...
aws_s3_bucket_object.finance-2020: Still creating...
[10s elapsed]
aws_s3_bucket.finance: Still creating... [10s
elapsed]
aws_s3_bucket_object.finance-2020: Still creating...
[20s elapsed]
aws_s3_bucket.finance: Still creating... [20s
```

>_ Terminal 2

\$ terraform apply

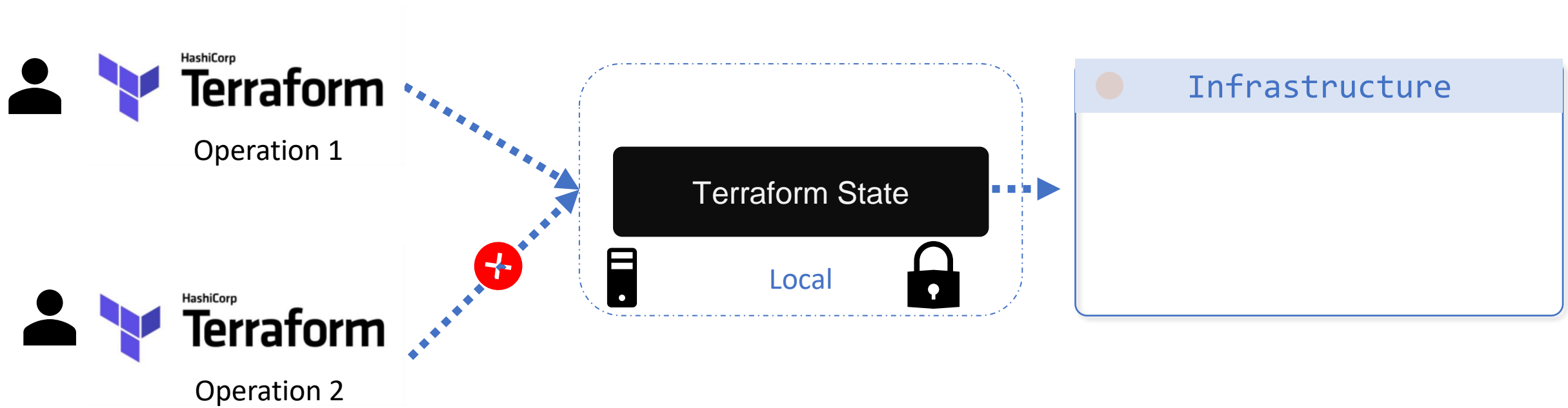
Error: Error locking state: Error acquiring the state
lock: resource temporarily unavailable

Lock Info:

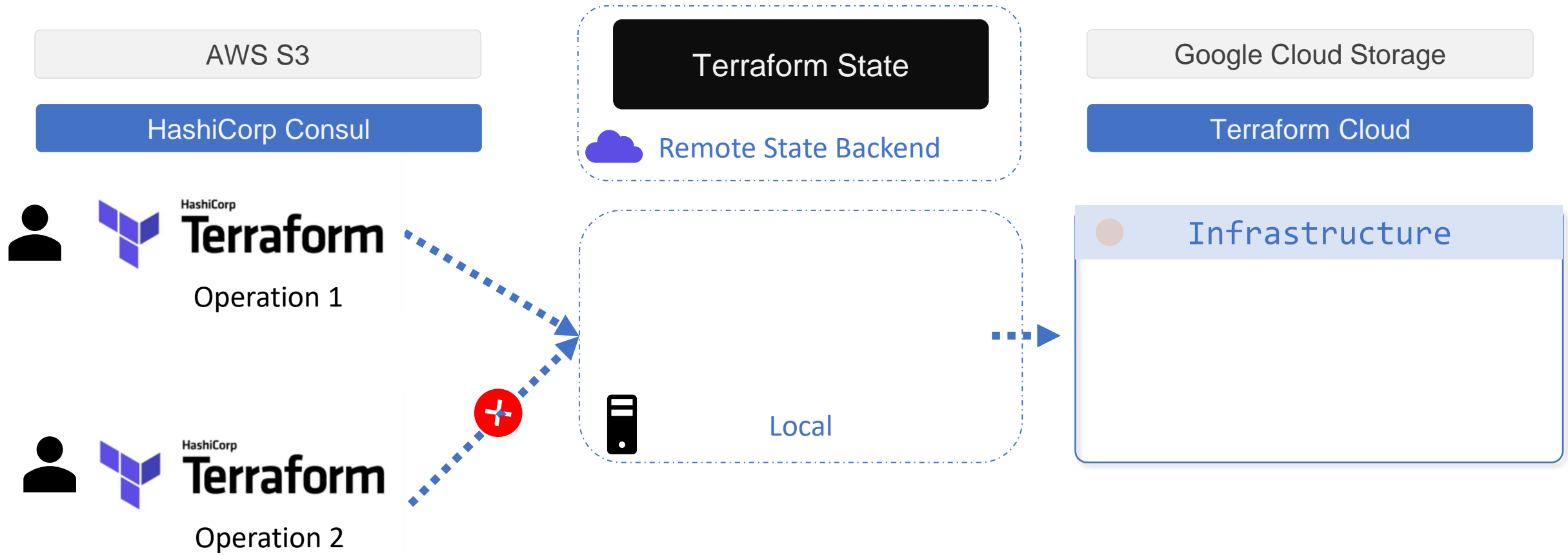
```
ID:          fefe3806-007c-084b-be61-cef4cdc77dee
Path:        terraform.tfstate
Operation:   OperationTypeApply
Who:         root@iac-server
Version:     0.13.3
Created:     2020-09-22 20:35:27.051330492 +0000 UTC
Info:
```

Terraform acquires a state lock to protect the state from
being written
by multiple users at the same time. Please resolve the
issue above and try
again. For most commands, you can disable locking with
the "-lock=false"
flag, but this is not recommended.

State Locking



State Locking



State Locking

AWS S3

HashiCorp Consul

Terraform State



Remote State Backend

Google Cloud Storage

Terraform Cloud

Automatically Load and Upload State
File

Many Backends Support State Locking

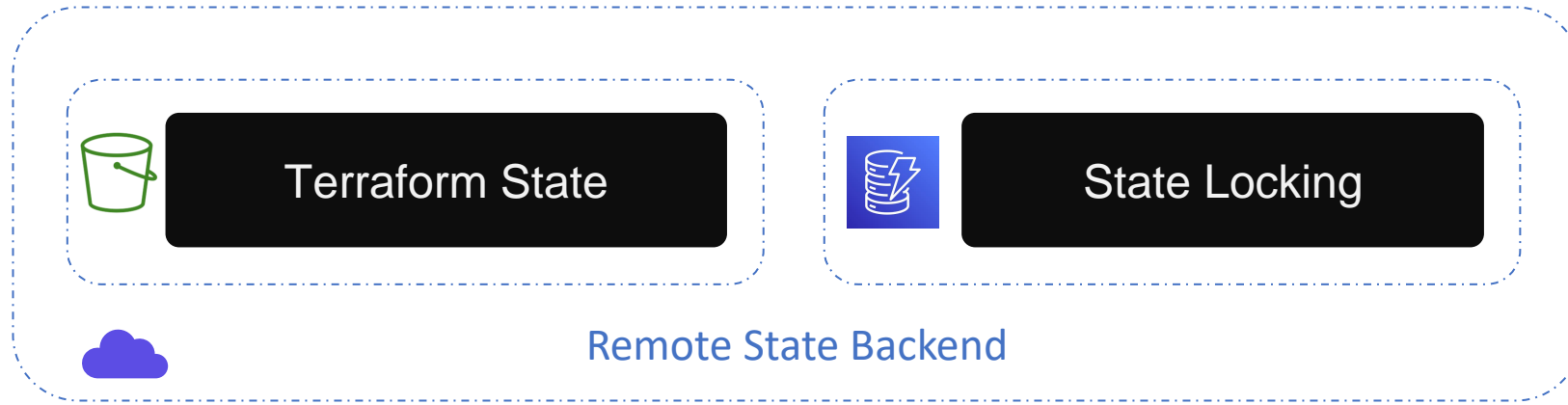
Security



{K}ODE{K}LOUD

Remote Backends with S3

Remote Backend



Object	Value
Bucket	kodecloud-terraform-state-bucket01
Key	finance/terraform.tfstate
Region	us-west-1
DynamoDB Table	state-locking

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}

terraform {
  backend "s3" {
    bucket      = "kodekloud-terraform-state-bucket01"
    key         = "finance/terraform.tfstate"
    region      = "us-west-1"
    dynamodb_table = "state-locking"
  }
}
```

> _

```
$ ls
main.tf  terraform.tfstate
```

Object	Value
Bucket	kodekloud-terraform-state-bucket01
Key	finance/terraform.tfstate
Region	us-west-1
DynamoDB Table	state-locking

main.tf

```
resource "local_file" "pet" {  
  filename = "/root/pets.txt"  
  content = "We love pets!"  
}
```

terraform.tf

```
terraform {  
  backend "s3" {  
    bucket      = "kodekloud-terraform-state-bucket01"  
    key         = "finance/terraform.tfstate"  
    region     = "us-west-1"  
    dynamodb_table = "state-locking"  
  }  
}
```

> _

\$ terraform apply

Backend reinitialization required. Please run "terraform init". Reason: Initial configuration of the requested backend "s3"

The "backend" is the interface that Terraform uses to store state, perform operations, etc. If this message is showing up, it means that the Terraform configuration you're using is using a custom configuration for the Terraform backend.

Changes to backend configurations require reinitialization. This allows Terraform to setup the new configuration, copy existing state, etc. This is only done during "terraform init". Please run that command now then try again.

Error: Initialization required. Please see the error message above.

> _

```
$ terraform init
```

Initializing the backend...

Do you want to copy existing state to the new backend?

Pre-existing state was found while migrating the previous "local" backend to the newly configured "s3" backend. No existing state was found in the newly configured "s3" backend. Do you want to copy this state to the new "s3" backend? Enter "yes" to copy and "no" to start with an empty state.

Enter a value: yes

Successfully configured the backend "s3"! Terraform will automatically use this backend unless the backend configuration changes.

Initializing provider plugins...

- Using previously-installed hashicorp/aws v3.7.0

.

.[Output Truncated]

> _

```
$ rm -rf terraform.tfstate
```

> _

\$ terraform apply

Acquiring state lock. This may take a few moments...

aws_s3_bucket.terraform-state: Refreshing state... [id=kodectloud-terraform-state-bucket01]

aws_dynamodb_table.state-locking: Refreshing state... [id=state-locking]

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.

Releasing state lock. This may take a few moments.



Terraform State Commands

```
> _  
  
$ vi terraform.tfstate ❌  
  
$ terraform state show aws_s3_bucket.finance ✅  
  
# terraform state <subcommand> [options] [args]
```

Sub-command
list
mv
pull
rm
show

```
terraform.tfstate  
  
{  
  "mode": "managed",  
  "type": "aws_instance",  
  "name": "dev-ec2",  
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",  
  "instances": [  
    {  
      "schema_version": 1,  
      "attributes": {  
        "ami": "ami-0a634ae95e11c6f91",  
        .  
        .  
        .  
        "primary_network_interface_id": "eni-0ccd57b1597e633e0",  
        "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",  
        "private_ip": "172.31.7.21",  
        "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",  
        "public_ip": "54.71.34.19",  
        "root_block_device": [  
          {  
            "delete_on_termination": true,  
            "device_name": "/dev/sda1",  
            "encrypted": false,  
            "iops": 100,  
            "kms_key_id": "",  
            "volume_id": "vol-070720a3636979c22",  
            "volume_size": 8,  
            "volume_type": "gp2"  
          }  
        ],  
      }  
    ],  
  }  
}
```


> _

```
# terraform state list [options] [address]
```

```
$ terraform state list
```

```
aws_dynamodb_table.cars
```

```
aws_s3_bucket.finance-2020922
```

```
$ terraform state list aws_s3_bucket.finance-2020922
```

```
aws_s3_bucket.finance-2020922
```

> _

```
# terraform state show [options] [address]
```

```
$ terraform state show aws_s3_bucket.finance-2020922
```

```
resource "aws_s3_bucket" "terraform-state" {
  acl                = "private"
  arn                 = "arn:aws:s3::: finance-2020922 "
  bucket              = "finance-2020922 "
  bucket_domain_name = "finance-2020922.s3.amazonaws.com"
  bucket_regional_domain_name = " finance-2020922.s3.us-west-1.amazonaws.com"
  force_destroy       = false
  hosted_zone_id      = "Z2F5ABCDE1ACD"
  id                  = "finance-2020922 "
  region              = "us-west-1"
  request_payer       = "BucketOwner"
  tags                = {
    "Description" = "Bucket to store Finance and Payroll Information"
  }

  versioning {
    enabled      = false
    mfa_delete   = false
  }
}
```

main.tf

```
resource "aws_dynamodb_table" "state-locking" {
  name = "state-locking-db"
  billing_mode = "PAY_PER_REQUEST"
  hash_key = "LockID"
  attribute {
    name = "LockID"
    type = "S"
  }
}
```

terraform.tfstate

```
"resources": [
  {
    "mode": "managed",
    "type": "aws_dynamodb_table",
    "name": "state-locking-db"
    "provider":
"provider[\"registry.terraform.io/hashicorp/aws\"]",
    .
    .
  }
]
```

> _

```
# terraform state mv [options] SOURCE DESTINATION
```

```
$ terraform state mv aws_dynamodb_table.state-locking aws_dynamodb_table.state-locking-db
```

```
Move "aws_dynamodb_table.state-locking" to "aws_dynamodb_table.state-locking-db"
```

```
Successfully moved 1 object(s).
```

```
$ terraform apply
```

```
aws_dynamodb_table.state-locking-db: Refreshing state... [id=state-locking]
```

```
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

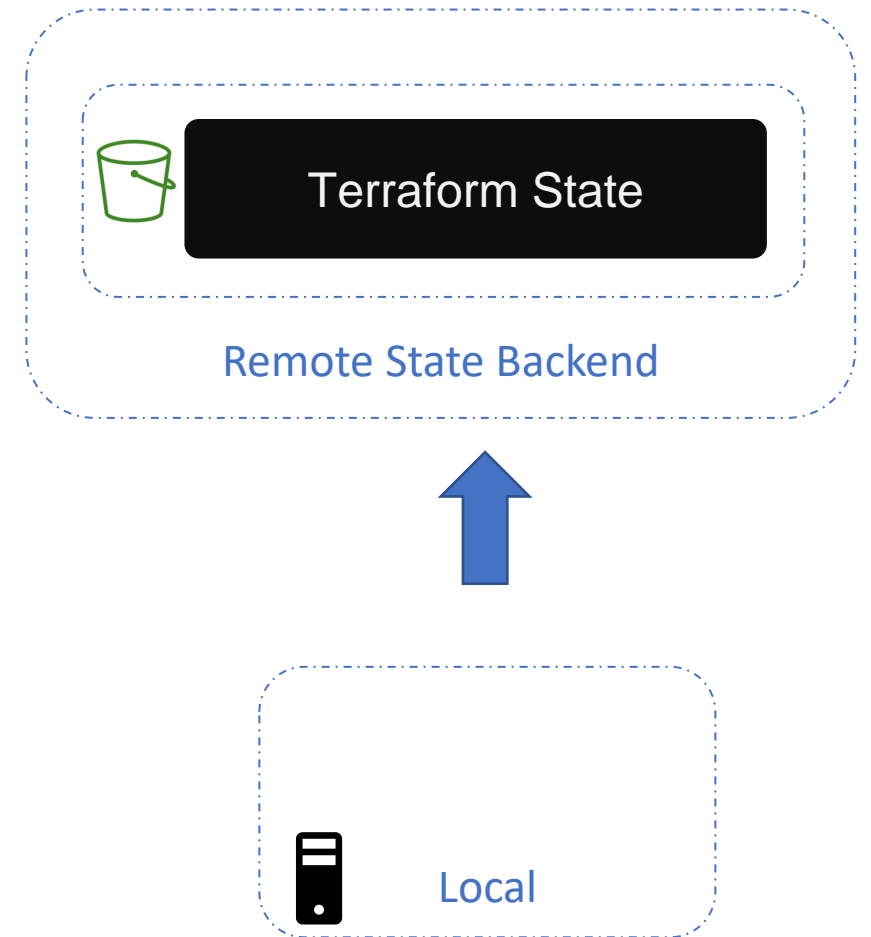
> _

```
$ ls
main.tf  provider.tf

# terraform state pull [options] SOURCE DESTINATION

$ terraform state pull
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 0,
  "lineage": "b6e2cf0e-ef8d-3c59-1e11-c6520dcd745c",
  "resources": [
    {
      "mode": "managed",
      "type": "aws_dynamodb_table",
      "name": "state-locking-db",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            ...

$ terraform state pull | jq '.resources[] | select(.name == "state-locking-
db")|.instances[].attributes.hash_key'
"LockID"
```



>_

```
# terraform state rm ADDRESS
```

```
$ terraform state rm aws_s3_bucket.finance-2020922
```

```
Acquiring state lock. This may take a few moments...
```

```
Removed aws_s3_bucket.finance-2020922
```

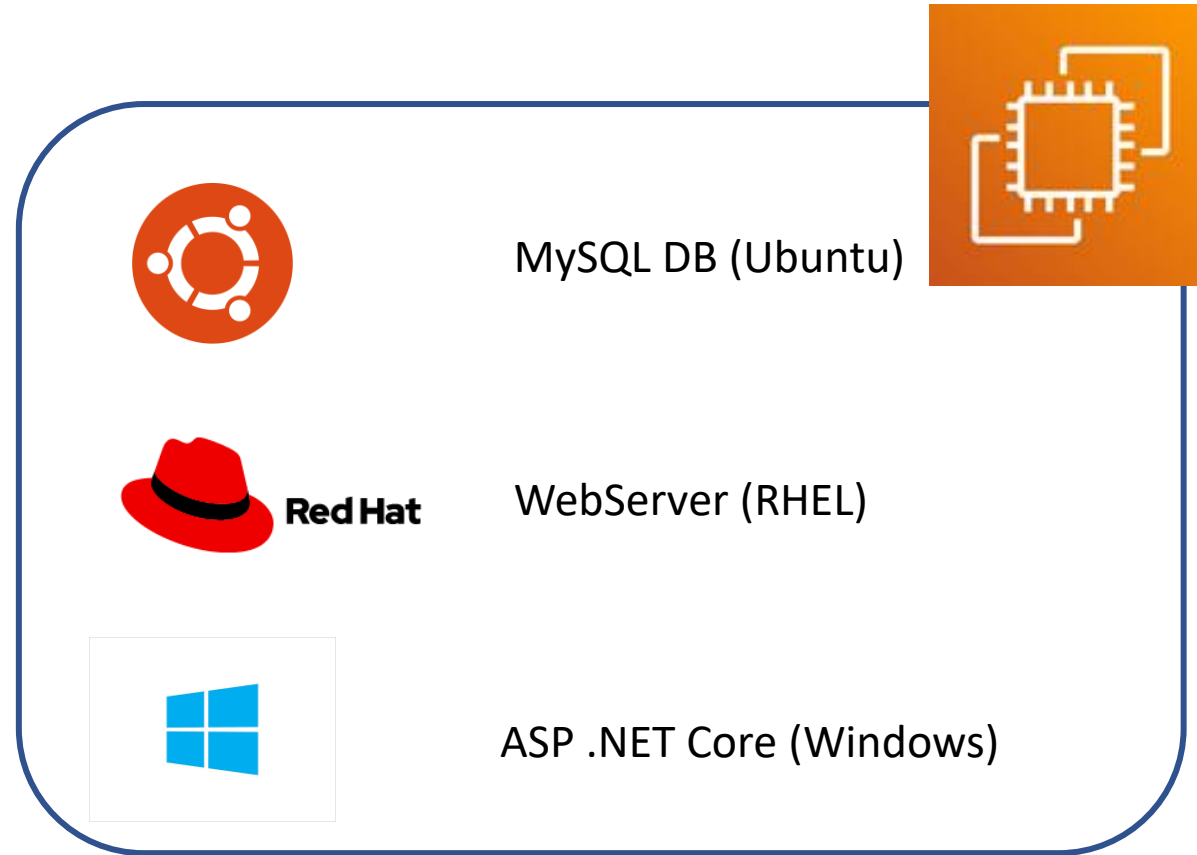
```
Successfully removed 1 resource instance(s).
```

```
Releasing state lock. This may take a few moments...
```



{K}ODE{K}LOUD

Introduction to AWS EC2



Elastic Compute Cloud

Amazon Machine Image (AMI's)



Amazon Linux 2 AMI ami-0c2f25c1f66a1ff4d

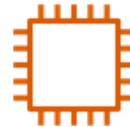


Red Hat Enterprise Linux 8 ami-04312317b9c8c4b51



Ubuntu Server 20.04 LTS ami-0edab43b6fa892279

Instance Types



General Purpose

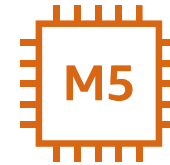
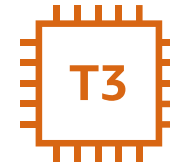
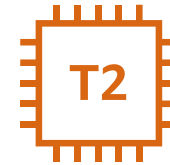


Compute Optimized



Memory Optimized

T2 General Purpose		
Instance Type	vCPU	Memory (GB)
t2.nano	1	0.5
t2.micro	1	1
t2.small	1	2
t2.medium	2	4
t2.large	2	8
t2.xlarge	4	16
t2.2xlarge	8	32

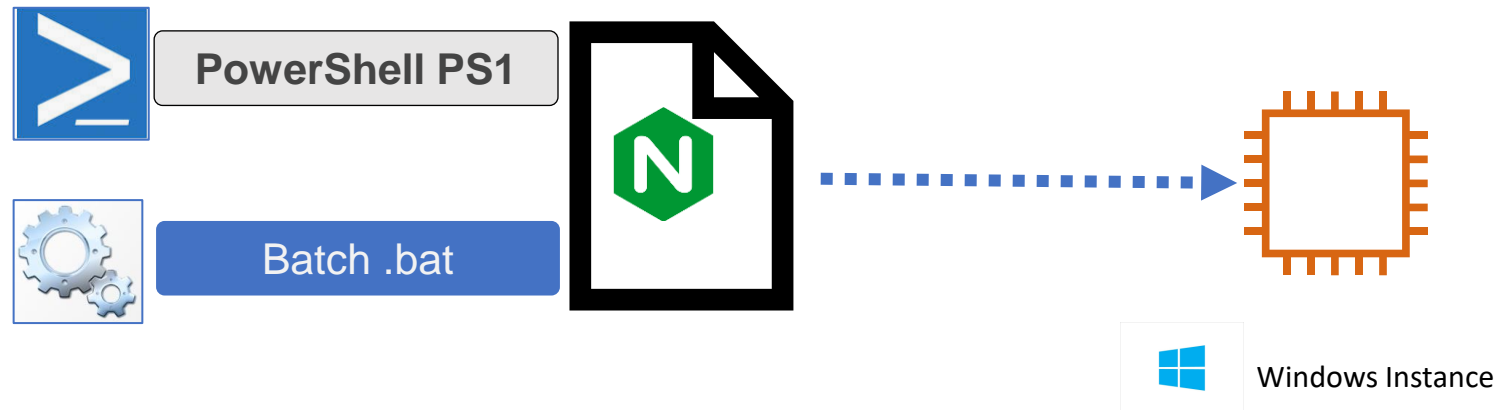
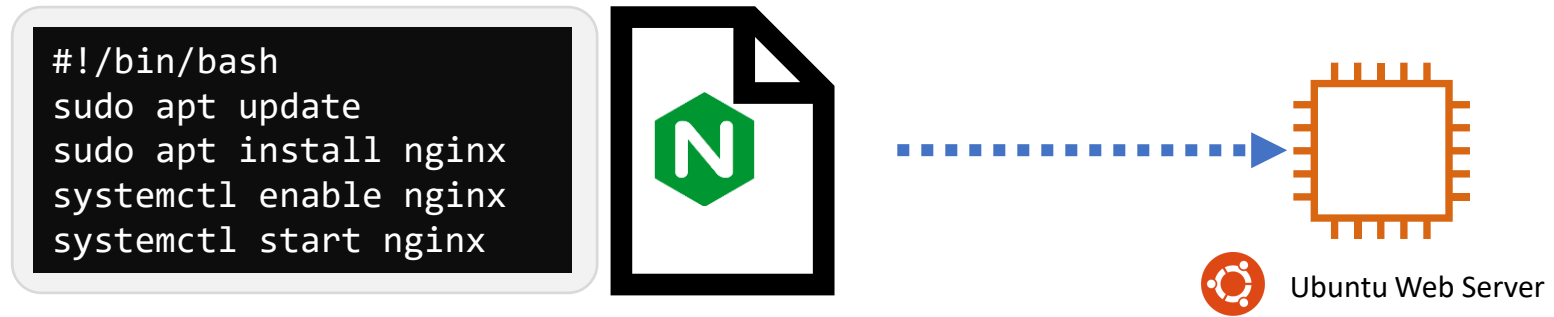


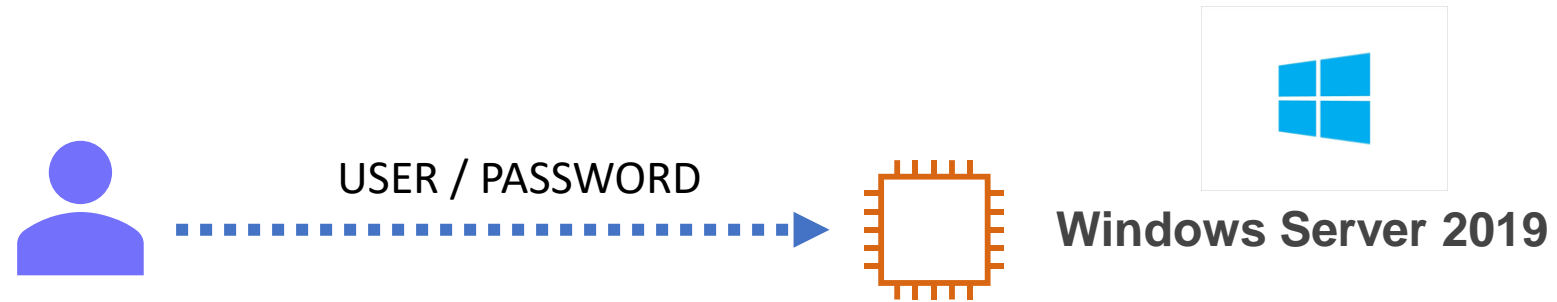
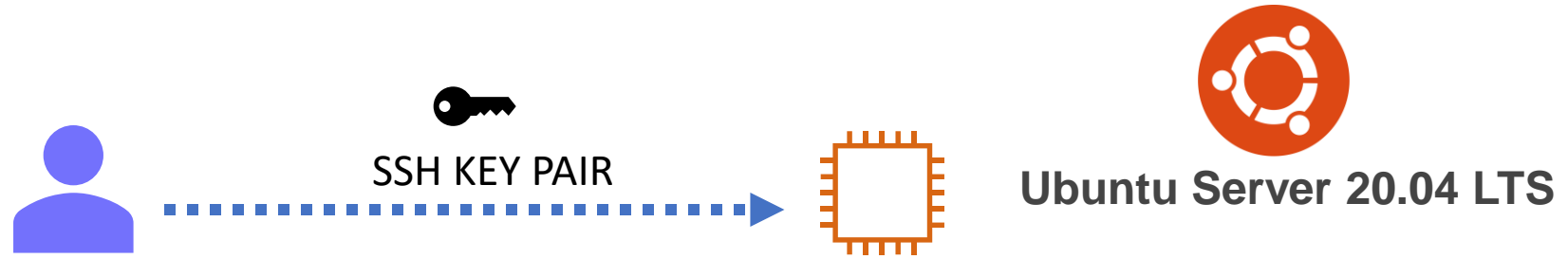


EBS Volume

EBS Volume Types		
Name	Type	Description
io1	SSD	For business-critical Apps
io2	SSD	For latency-sensitive transactional workloads
gp2	SSD	General Purpose
st1	HDD	Low Cost HDD frequently accessed, throughput-intensive workloads
sc1	HDD	Lowest cost HDD volume designed for less frequently accessed workloads

User Data







AWS EC2 With Terraform

main.tf

```
resource "aws_instance" "webserver" {  
    ami          = "ami-0edab43b6fa892279"  
    instance_type = "t2.micro"  
}
```

Argument Reference

The following arguments are supported:

- `ami` - (Required) The AMI to use for the instance.
- `instance_type` - (Required) The type of instance to start. Updates to this field will trigger a stop/start of the EC2 instance.
- `tags` - (Optional) A map of tags to assign to the resource.

provider.tf

```
provider "aws" "  
    region = "us-west-1"  
}
```


main.tf

```
resource "aws_instance" "webserver" {  
    ami          = "ami-0edab43b6fa892279"  
    instance_type = "t2.micro"  
    tags = {  
        Name          = "webserver"  
        Description = "An Nginx WebServer on Ubuntu"  
    }  
}
```

Argument Reference

The following arguments are supported:

- `ami` - (Required) The AMI to use for the instance.
- `instance_type` - (Required) The type of instance to start. Updates to this field will trigger a stop/start of the EC2 instance.
- `tags` - (Optional) A map of tags to assign to the resource.

provider.tf

```
provider "aws" {  
    region = "us-west-1"  
}
```

main.tf

```
resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
}
```

provider.tf

```
provider "aws" " " {
  region = "us-west-1"
}
```

Argument Reference

The following arguments are supported:

- `ami` - (Required) The AMI to use for the instance.
- `instance_type` - (Required) The type of instance to start. Updates to this field will trigger a stop/start of the EC2 instance.
- `tags` - (Optional) A map of tags to assign to the resource.
- `user_data` - (Optional) The user data to provide when launching the instance. Do not pass gzip-compressed data via this argument; see `user_data_base64` instead.

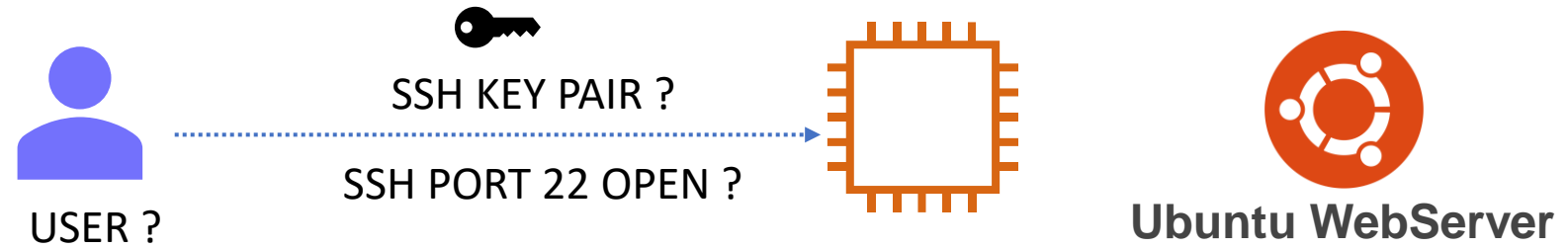
> _

\$ terraform apply

```
# aws_instance.webserver will be created
+ resource "aws_instance" "webserver" {
  + ami                                = "ami-0edab43b6fa892279"
  .
  .
  + instance_type                     = "t2.micro"
  + ipv6_address_count                = (known after apply)
  + public_ip                         = (known after apply)
  + source_dest_check                 = true
  + subnet_id                         = (known after apply)
  + tags                             = {
    + "Description" = "An NGINX WebServer on Ubuntu"
    + "Name"        = "webserver"
  }
  + tenancy                         = (known after apply)
  + user_data                       = "527516162d9d8675a26b6ca97664226e6e2bff82"
  + volume_tags                     = (known after apply)
  + vpc_security_group_ids          = (known after apply)
  .
  .
aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Creation complete after 22s [id=i-0085e5d0f442f7c4f]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

<input type="checkbox"/>	Name ▾	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm Status	Availability zone ▾
<input type="checkbox"/>	webserver	i-0085e5d0f442f7c4f	✔ Running	t2.micro	✔ 2/2 checks ...	No alarms +	ca-central-1a



Select an existing key pair or create a new key pair ✕

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Create a new key pair ▼

Key pair name

webserver

Download Key Pair



You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Cancel

Launch Instances

main.tf

```
resource "aws_instance" "webserver" {  
  ami          = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name          = "webserver"  
    Description = "An Nginx WebServer on Ubuntu"  
  }  
  user_data = <<-EOF  
    #!/bin/bash  
    sudo apt update  
    sudo apt install nginx -y  
    systemctl enable nginx  
    systemctl start nginx  
    EOF  
}
```

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
}

resource "aws_key_pair" "web" {
  public_key = file("/root/.ssh/web.pub")
}
```

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An NGINX WebServer on Ubuntu"
  }
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
}

resource "aws_key_pair" "web" {
  public_key = "ssh-
rsa AAAAB3NzaC1yc2EAAAADAQABAAQDdicpU+kT9isaZy7cHYa
+oCTUolS6Tg6vCEq+ufucIMrA7RLTngi+YfTfvgrY2UiHGxuuJ1lE
yT0x2UrGexVx4G2TzX/am2WFzNbcGSg2bCXTkVQY93K0hbW9y851a
+g1wI7TODC0oxEMFr/CVsrJ4bfbp8S896VKBxC1WpSU9GscPP28GV
uDgm2ATBuL78AF root@iac-server"
}
```

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
  key_name = aws_key_pair.web.id
}

resource "aws_key_pair" "web" {
  public_key = file("/root/.ssh/web.pub")
}
```

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the low

Number of instances ⓘ [Launch into Auto Scaling Group ⓘ](#)

Purchasing option ⓘ ☐ Request Spot instances

Network ⓘ

vpc-7da8d215 (default) ⌵ [Create new VPC](#)

Subnet ⓘ

No preference (default subnet in any Availability Zone) ⌵ [Create new subnet](#)

Auto-assign Public IP ⓘ

Use subnet setting (Enable) ⌵

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group
☐ Select an existing security group

Security group name:

Description:

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
SSH ⌵	TCP	22	Custom ⌵ 0.0.0.0/0

Add Rule

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF

  key_name = aws_key_pair.web.id
}

resource "aws_key_pair" "web" {
  public_key = file("/root/.ssh/web.pub")
}
```

```

    }
    user_data = <<-EOF
        #!/bin/bash
        sudo apt update
        sudo apt install nginx -y
        systemctl enable nginx
        systemctl start nginx
    EOF

    key_name = aws_key_pair.web.id
}

resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}

resource "aws_security_group" "ssh-access" {
    name          = "ssh-access"
    description   = "Allow SSH access from the Internet"
    ingress {
        from_port   = 22
        to_port     = 22
        protocol     = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
    }
}

```

The `ingress` block supports:

- `cidr_blocks` - (Optional) List of CIDR blocks.
- `to_port` - (Required) The end range port (or ICMP code if protocol is "icmp")
- `from_port` - (Required) The start port (or ICMP type number if protocol is "icmp" or "icmpv6")
- `protocol` - (Required) The protocol. If you select a protocol of "-1" (semantically equivalent to `"all"`, which is not a valid value here), you must specify a `"from_port"` and `"to_port"` equal to 0. If not icmp, icmpv6, tcp, udp, or "-1" use the `protocol`

```
user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
EOF

key_name = aws_key_pair.web.id
vpc_security_group_ids = [aws_security_group.ssh-access.id ]
}

resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}

resource "aws_security_group" "ssh-access" {
    name = "ssh-access"
    description = "AllowSSH access from the Internet"
    ingress {
        from_port = 22
        to_port = 22
        protocol = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
    }
}
```

```
        systemctl start nginx
    EOF

    key_name    = aws_key_pair.web.id
    vpc_security_group_ids = [ aws_security_group.ssh-access.id ]

}

resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}

resource "aws_security_group" "ssh-access" {
    name            = "ssh-access"
    description     = "AllowSSH access from the Internet"
    ingress {
        from_port    = 22
        to_port      = 22
        protocol      = "tcp"
        cidr_blocks  = ["0.0.0.0/0"]
    }
}

output publicip {
    value = aws_instance.webserver.public_ip
}
```

>
_

```
$ terraform apply
```

```
Plan: 3 to add, 0 to change, 1 to destroy.
```

```
Do you want to perform these actions?
```

```
Terraform will perform the actions described above.
```

```
Only 'yes' will be accepted to approve.
```

```
Enter a value: yes
```

```
aws_instance.webserver: Destroying... [id=i-015b579d0ea84fbb7]
```

```
aws_key_pair.web: Creating...
```

```
aws_security_group.ssh-access: Creating...
```

```
aws_key_pair.web: Creation complete after 1s [id=terraform-  
20201014034144926200000001]
```

```
aws_security_group.ssh-access: Creation complete after 1s [id=sg-  
0f02f3ea92b14bed8]
```

```
aws_instance.webserver: Still destroying... [id=i-015b579d0ea84fbb7, 10s elapsed]
```

```
aws_instance.webserver: Still destroying... [id=i-015b579d0ea84fbb7, 20s elapsed]
```

```
aws_instance.webserver: Destruction complete after 30s
```

```
aws_instance.webserver: Creating...
```

```
aws_instance.webserver: Still creating... [10s elapsed]
```

```
aws_instance.webserver: Still creating... [20s elapsed]
```

```
aws_instance.webserver: Still creating... [30s elapsed]
```

```
aws_instance.webserver: Creation complete after 32s [id=i-0fd2c1c5eb0762ff5]
```

```
Apply complete! Resources: 3 added, 0 changed, 1 destroyed.
```

```
Outputs:
```

```
publicip = 3.96.203.171
```

>
_

```
$ ssh -i /root/.ssh/web ubuntu@3.96.203.171
```

```
ubuntu@ip-172-31-19-161:~$
```

```
[ubuntu@ip-172-31-19-161]$ systemctl status nginx
```

```
nginx.service - A high performance web server and a reverse proxy server
```

```
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
```

```
   Active: active (running) since Wed 2020-11-02 22:17:38 UTC; 2 min ago
```

```
   Process: 303 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0
```

```
   Process: 264 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited,
```

```
   Main PID: 304 (nginx)
```



{K}ODE{K}LOUD

Terraform Provisioners

Provisioners

main.tf

```
resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
  key_name     = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]
}

resource "aws_key_pair" "web" {
  << code hidden >>
}

resource "aws_security_group" "ssh-access" {
  << code hidden >>
}
```

▼ Advanced Details

Metadata accessible



Enabled

Metadata version



V1 and V2 (token optional)

Metadata token response hop limit



1

User data



☒ As text ☐ As file ☐ Input is already base64 encoded

```
#!/bin/bash
sudo apt update
sudo apt install nginx -y
systemctl enable nginx
systemctl start nginx
```

Remote Exec

main.tf

```
resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
  key_name     = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]
}

resource "aws_key_pair" "web" {
  << code hidden >>
}

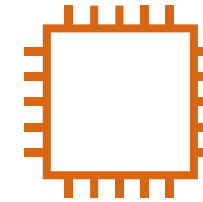
resource "aws_security_group" "ssh-access" {
  << code hidden >>
}
```

```
apt update
apt install nginx -y
systemctl enable nginx
systemctl start nginx
```

Remote Instance (EC2)



SSH



WINRM



HashiCorp
Terraform

Local Machine



- ✓ Network Connectivity (Security Group)
- ✓ Authentication (SSH Key Pair)

Remote Exec

```
main.tf

resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
    inline = [ "sudo apt update",
               "sudo apt install nginx -y",
               "sudo systemctl enable nginx",
               "sudo systemctl start nginx",
             ]
  }
  key_name      = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]
}

resource "aws_key_pair" "web" {
  << code hidden >>
}

resource "aws_security_group" "ssh-access" {
  << code hidden >>
}
```

Remote Exec

main.tf

```
resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
    inline = [ "sudo apt update",
              "sudo apt install nginx -y",
              "sudo systemctl enable nginx",
              "sudo systemctl start nginx",
            ]
  }
  connection {
    type        = "ssh"
    host        = self.public_ip
    user        = "ubuntu"
    private_key = file("/root/.ssh/web")
  }
  key_name     = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]
}

resource "aws_key_pair" "web" {
  << code hidden >>
}
```

>_

```
$ terraform apply
aws_key_pair.web: Creating...
aws_security_group.ssh-access: Creating...
aws_key_pair.web: Creation complete after 0s [id=terraform-20201015013048509100000001]
aws_security_group.ssh-access: Creation complete after 1s [id=sg-0]
aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [10s elapsed]
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Still creating... [30s elapsed]
aws_instance.webserver: Provisioning with 'remote-exec'...
aws_instance.webserver (remote-exec): Connecting to remote host via ssh...
aws_instance.webserver (remote-exec): Host: 3.96.136.157
aws_instance.webserver (remote-exec): User: ubuntu
aws_instance.webserver (remote-exec): Password: false
aws_instance.webserver (remote-exec): Private key: true
aws_instance.webserver (remote-exec): Certificate: false
aws_instance.webserver (remote-exec): SSH Agent: false
aws_instance.webserver (remote-exec): Checking Host Key: false
aws_instance.webserver: Still creating... [40s elapsed]
aws_instance.webserver (remote-exec): Connecting to remote host via ssh...
aws_instance.webserver (remote-exec): Host: 3.96.136.157
aws_instance.webserver (remote-exec): User: ubuntu
aws_instance.webserver (remote-exec): Password: false
aws_instance.webserver (remote-exec): Private key: true
aws_instance.webserver (remote-exec): Certificate: false
aws_instance.webserver (remote-exec): SSH Agent: false
aws_instance.webserver (remote-exec): Checking Host Key: false
aws_instance.webserver (remote-exec): Connected!
aws_instance.webserver: Still creating... [50s elapsed]
aws_instance.webserver: Creation complete after 50s [id=i-068fad30]
```

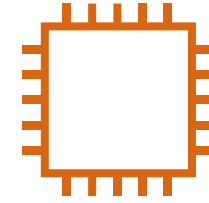
Local Exec

```
main.tf

resource "aws_instance" "webserver" {
  ami          = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
    inline = [ "sudo apt update",
               "sudo apt install nginx -y",
               "sudo systemctl enable nginx",
               "sudo systemctl start nginx",
             ]
  }
  connection {
    type        = "ssh"
    host        = self.public_ip
    user        = "ubuntu"
    private_key = file("/root/.ssh/web")
  }
  key_name     = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]
}

resource "aws_key_pair" "web" {
  << code hidden >>
}
```

Remote Instance (EC2)



Local Machine



```
echo ${aws_instance.webserver.public_ip} >> /tmp/ip.txt"
```

Local Exec

main.tf

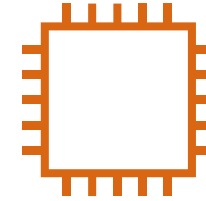
```
resource "aws_instance" "webserver" {  
  ami          = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  
  provisioner "local-exec" {  
    command = "echo ${aws_instance.webserver2.public_ip} >> /tmp/ips.txt"  
  }  
}
```

> _

```
$ cat /tmp/ips.txt  
54.214.68.27
```

- **command** - (Required) This is the command to execute. It can be provided as a relative path to the current working directory or as an absolute path. It is evaluated in a shell, and can use environment variables or Terraform variables.

Remote Instance (EC2)



Local Machine



```
echo ${aws_instance.webserver.public_ip} >> /tmp/ip.txt"
```



{K}ODE{K}LOUD

Provisioner Behavior

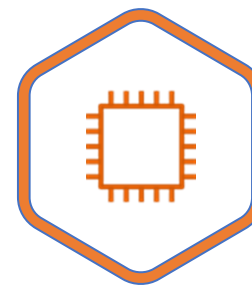
Creation Time Provisioner

main.tf

```
resource "aws_instance" "webserver" {  
  ami           = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  provisioner "local-exec" {  
    command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /tmp/instance_state.txt"  
  }  
}
```

>_

```
$ cat /tmp/instance_state.txt  
Instance 3.96.136.157 Created!
```



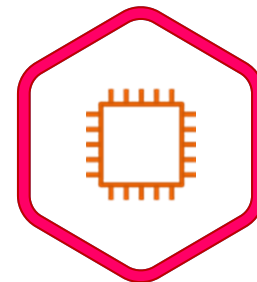
Destroy Time Provisioner

main.tf

```
resource "aws_instance" "webserver" {  
  ami           = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  provisioner "local-exec" {  
    command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /tmp/instance_state.txt"  
  }  
  provisioner "local-exec" {  
    when      = destroy  
    command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! > /tmp/instance_state.txt"  
  }  
}
```

> _

```
$ cat /tmp/instance_state.txt  
Instance 3.96.136.157 Deleted!
```



Failure Behavior

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "local-exec" {
    on_failure = fail
    command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /temp/instance_state.txt"
  }
  provisioner "local-exec" {
    when      = destroy
    command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! > /tmp/instance_state.txt"
  }
}
```

> _

\$ terraform apply

Error: Error running command 'echo 35.183.14.192 > /temp/pub_ip.txt': exit status 1.
Output: The system cannot find the path specified.

Failure Behavior

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "local-exec" {
    on_failure = continue
    command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /temp/instance_state.txt"
  }
  provisioner "local-exec" {
    when      = destroy
    command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! > /tmp/instance_state.txt"
  }
}
```

> _

```
$ terraform apply
aws_instance.webserver (local-exec) The system cannot find the path specified.
aws_instance.project: Creation complete after 22s [id=i-01585c2b9dbc445db]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```



Considerations with Provisioners

Local-Exec | Remote-Exec

main.tf

```
resource "aws_instance" "webserver" {  
  ami = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "webserver"  
    Description = "An NGINX WebServer on Ubuntu"  
  }  
  provisioner "remote-exec" {  
    inline = ["echo $(hostname -i) >> /tmp/ips.txt"]  
  }  
}
```

No Provisioner Information in Plan

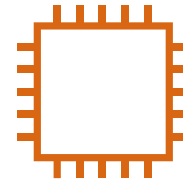
Network Connectivity and Authentication

Local Machine



SSH/ WinRM

EC2 Instance



main.tf

```
resource "aws_instance" "webserver" {
  ami = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name = "webserver"
    Description = "An NGINX WebServer on Ubuntu"
  }
  user_data = <<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
  EOF
}
```

Provider	Resource	Option
AWS	aws_instance	user_data
Azure	azurerm_virtual_machine	custom_data
GCP	google_compute_instance	meta_data
Vmware vSphere	vsphere_virtual_machine	user_data.txt

main.tf

```
resource "aws_instance" "webserver" {  
  ami = "ami-XYZ"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "webserver"  
    Description = "An NGINX WebServer on Ubuntu"  
  }  
}
```



Custom AMI with NGINX



nginx-build.json



Terraform Taint

Taint

main.tf

```
resource "aws_instance" "webserver-3" {  
  ami           = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  key_name      = "ws"  
  provisioner "local-exec" {  
    command = "echo ${aws_instance.webserver-3.public_ip} > /temp/pub_ip.txt"  
  }  
}
```

> _

\$ terraform apply

Plan: 1 to add, 0 to change, 0 to destroy.

```
aws_instance.webserver: Creating...  
aws_instance.webserver: Still creating... [10s elapsed]  
aws_instance.webserver: Still creating... [20s elapsed]  
aws_instance.webserver: Still creating... [30s elapsed]  
aws_instance.webserver: Provisioning with 'local-exec'...  
aws_instance.webserver (local-exec): Executing: ["cmd" "/C" "echo 35.183.14.192 > /temp/pub_ip.txt"]  
aws_instance.webserver (local-exec): The system cannot find the path specified.
```

Error: Error running command 'echo 35.183.14.192 > /temp/pub_ip.txt': exit status 1. Output: The system cannot find the path specified.

Taint

> _

```
$ terraform plan
```

```
Refreshing Terraform state in-memory prior to plan...
```

```
The refreshed state will be used to calculate this plan, but will not  
be  
persisted to local or remote state storage.
```

```
aws_instance.webserver: Refreshing state... [id=i-0dba2d5dc22a9a904]
```

```
-----  
-
```

```
An execution plan has been generated and is shown below.
```

```
Resource actions are indicated with the following symbols:
```

```
-/+ destroy and then create replacement
```

```
Terraform will perform the following actions:
```

```
  # aws_instance.webserver is tainted, so must be replaced  
-/+ resource "aws_instance" "webserver-3" {
```

Taint

> _

```
$ terraform taint aws_instance.webserver
Resource instance aws_instance.webserver has been marked as tainted.

$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

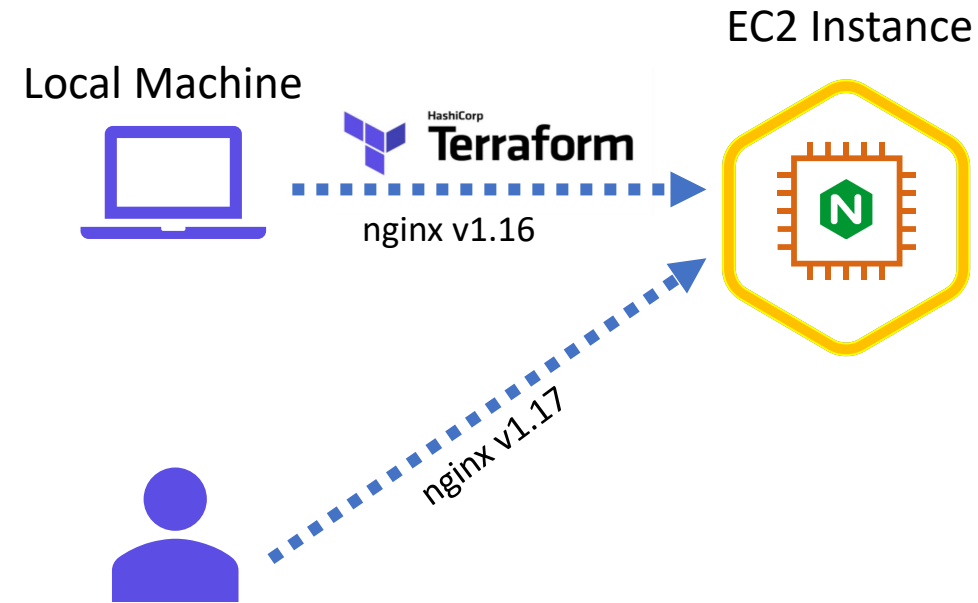
aws_instance.webserver: Refreshing state... [id=i-0fd3946f5b3ab8af8]

-----

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

# aws_instance.webserver is tainted, so must be replaced
-/+ resource "aws_instance" "webserver" {
```



Taint

> _

```
$ terraform untaint aws_instance.webserver
```

```
Resource instance aws_instance.webserver has been successfully  
untainted.
```

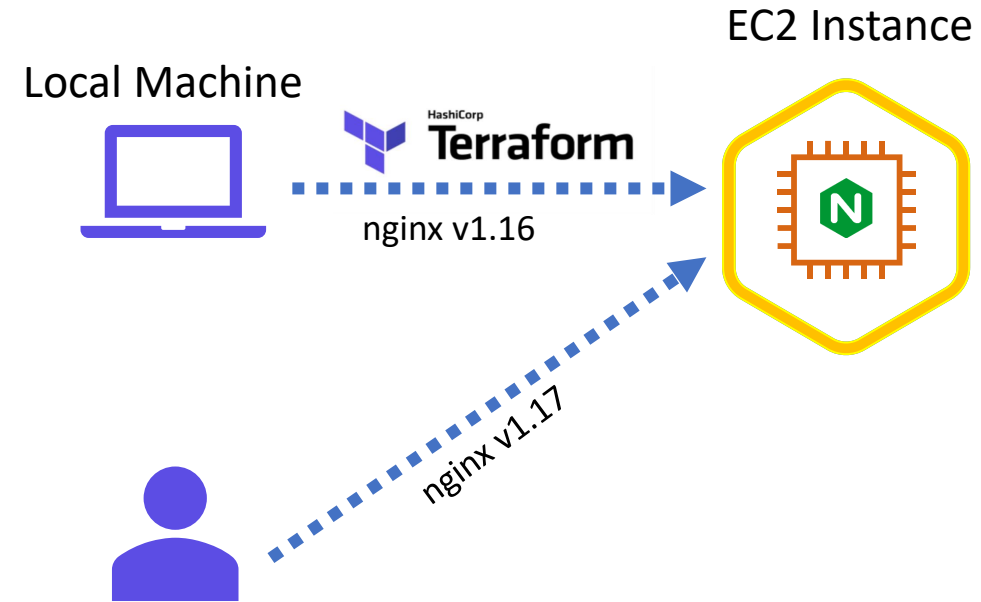
```
$ terraform plan
```

```
Refreshing Terraform state in-memory prior to plan...  
The refreshed state will be used to calculate this plan, but will not be  
persisted to local or remote state storage.
```

```
aws_instance.webserver: Refreshing state... [id=i-0fd3946f5b3ab8af8]
```

```
-----  
No changes. Infrastructure is up-to-date.
```

```
This means that Terraform did not detect any differences between your  
configuration and real physical resources that exist. As a result, no  
actions need to be performed.
```





Debugging

Log Levels

```
>_
```

```
# export TF_LOG=<log_level>
```

```
$ export TF_LOG=TRACE
```

INFO

WARNING

ERROR

DEBUG

TRACE

> _

\$ terraform plan

```
----
2020/10/18 22:08:30 [INFO] Terraform version: 0.13.0
2020/10/18 22:08:30 [INFO] Go runtime version: go1.14.2
2020/10/18 22:08:30 [INFO] CLI args: []string{"C:\\Windows\\system32\\terraform.exe", "plan"}
2020/10/18 22:08:30 [DEBUG] Attempting to open CLI config file: C:\\Users\\vpala\\AppData\\Roaming\\terraform.rc
2020/10/18 22:08:30 [DEBUG] File doesn't exist, but doesn't need to. Ignoring.
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory terraform.d/plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory C:\\Users\\vpala\\AppData\\Roaming\\terraform.d\\plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\\Users\\vpala\\AppData\\Roaming\\HashiCorp\\Terraform\\plugins
2020/10/18 22:08:30 [INFO] CLI command args: []string{"plan"}
2020/10/18 22:08:30 [WARN] Log levels other than TRACE are currently unreliable, and are supported only for backward
compatibility.
    Use TF_LOG=TRACE to see Terraform's internal logs.
----
2020/10/18 22:08:30 [DEBUG] New state was assigned lineage "f413959c-538a-f9ce-524e-1615073518d4"
2020/10/18 22:08:30 [DEBUG] checking for provisioner in "."
2020/10/18 22:08:30 [DEBUG] checking for provisioner in "C:\\Windows\\system32"
2020/10/18 22:08:30 [INFO] Failed to read plugin lock file .terraform\\plugins\\windows_amd64\\lock.json: open
.terraform\\plugins\\windows_amd64\\lock.json: The system cannot find the path specified.
2020/10/18 22:08:30 [INFO] backend/local: starting Plan operation
2020-10-18T22:08:30.625-0400 [INFO] plugin: configuring client automatic mTLS
2020-10-18T22:08:30.646-0400 [DEBUG] plugin: starting plugin:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
args=[.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe]
2020-10-18T22:08:30.935-0400 [DEBUG] plugin: plugin started:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
pid=34016
2020-10-18T22:08:30.935-0400 [DEBUG] plugin: waiting for RPC address:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
2020-10-18T22:08:30.974-0400 [INFO] plugin.terraform-provider-aws_v3.11.0_x5.exe: configuring server automatic mTLS:
```

> _

```
$ export TF_LOG_PATH=/tmp/terraform.log
```

```
$ head -10 /tmp/terraform.log
```

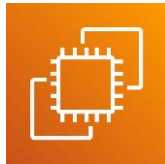
```
----  
2020/10/18 22:08:30 [INFO] Terraform version: 0.13.0  
2020/10/18 22:08:30 [INFO] Go runtime version: go1.14.2  
2020/10/18 22:08:30 [INFO] CLI args: []string{"C:\\Windows\\system32\\terraform.exe",  
"plan"}  
2020/10/18 22:08:30 [DEBUG] Attempting to open CLI config file:  
C:\\Users\\vpala\\AppData\\Roaming\\terraform.rc  
2020/10/18 22:08:30 [DEBUG] File doesn't exist, but doesn't need to. Ignoring.  
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory  
terraform.d/plugins  
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory  
C:\\Users\\vpala\\AppData\\Roaming\\terraform.d\\plugins  
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory  
C:\\Users\\vpala\\AppData\\Roaming\\HashiCorp\\Terraform\\plugins  
2020/10/18 22:08:30 [INFO] CLI command args: []string{"plan"}
```

```
$ unset TF_LOG_PATH
```

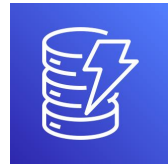


{K}ODE{K}LOUD

Terraform Import



EC2



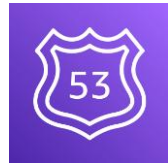
DynamoDB



Elastic Block Store



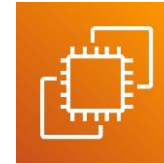
S3



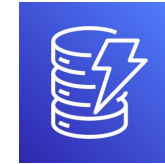
Route 53



VPC



EC2



DynamoDB



Route 53

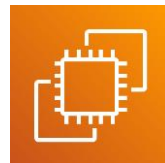
AWS Management Console



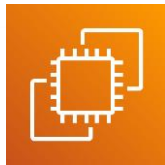
S3



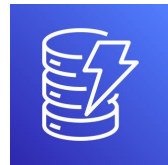
Elastic Block Store



EC2



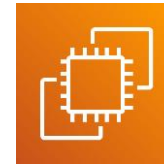
EC2



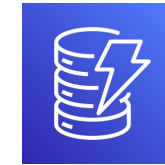
DynamoDB



Elastic Block Store



EC2



DynamoDB



Route 53



S3



Route 53



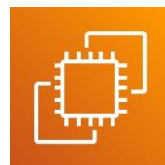
VPC



S3



Elastic Block Store



EC2

Data Source

main.tf

```
.  
.  
data "aws_instance" "newserver" {  
    instance_id = "i-026e13be10d5326f7"  
}  
output newserver {  
    value      = data.aws_instance.newserver.public_ip  
}
```

>_

```
$ terraform apply  
  
$ data.aws_instance.newserver: Refreshing state... [id=i-026e13be10d5326f7]  
aws_key_pair.web: Refreshing state... [id=terraform-20201015013048509100000001]  
aws_security_group.ssh-access: Refreshing state... [id=sg-0a543f25009e14628]  
aws_instance.webserver: Refreshing state... [id=i-068fad300d9df27ac]  
  
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.  
  
Outputs:  
  
newserver = 15.223.1.176
```

Terraform Import

> _

```
# terraform import <resource_type>.<resource_name> <attribute>
```

```
$ terraform import aws_instance.webserver-2 i-026e13be10d5326f7
```

Error: resource address "aws_instance.webserver-2" does not exist in the configuration.

Before importing this resource, please create its configuration in the root module. For example:

```
resource "aws_instance" "webserver-2" {  
  # (resource arguments)  
}
```

Terraform Import

main.tf

```
resource "aws_instance" "webserver-2" {  
  # (resource arguments)  
}
```

```
>_  
  
$ terraform import aws_instance.webserver-2 i-026e13be10d5326f7  
  
aws_instance.webserver-2: Importing from ID "i-026e13be10d5326f7"...  
aws_instance.webserver-2: Import prepared!  
  Prepared aws_instance for import  
aws_instance.webserver-2: Refreshing state... [id=i-026e13be10d5326f7]  
  
Import successful!  
  
The resources that were imported are shown above. These resources are now in  
your Terraform state and will henceforth be managed by Terraform.
```

Instance summary for i-0d7c0088069819ff8 (old-ec2) [Info](#)

Updated less than a minute ago



Connect

Actions

Instance ID

i-0d7c0088069819ff8 (old-ec2)

Instance state

Running

Instance type

t2.micro

IAM Role

—

Public IPv4 address

15.223.5.69 | [open address](#)

Public IPv4 DNS

ec2-15-223-5-69.ca-central-1.compute.amazonaws.com | [open address](#)

Elastic IP addresses

—

Subnet ID

subnet-c6c0a8ae

Private IPv4 addresses

172.31.23.147

Private IPv4 DNS

ip-172-31-23-147.ca-central-1.compute.internal

VPC ID

vpc-7da8d215



AWS Compute Optimizer

Opt-in to AWS Compute Optimizer for recommendations.

[Learn more](#)

Details

Security

Networking

Storage

Monitoring

Tags

▼ Instance details [Info](#)

Platform

Ubuntu (Inferred)

AMI ID

ami-0edab43b6fa892279

Monitoring

disabled

terraform.tfstate

```
{
  "mode": "managed",
  "type": "aws_instance",
  "name": "webserver-2",
  "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
  "instances": [
    {
      "schema_version": 1,
      "attributes": {
        "ami": "ami-0edab43b6fa892279",
        "instance_state": "running",
        "instance_type": "t2.micro",
        "key_name": "ws",
        "tags": {
          "Name": "old-ec2"
        }
      },
      "vpc_security_group_ids": [
        "sg-8064fdee"
      ]
    }
  ]
}
```

main.tf

```
resource "aws_instance" "webserver-2" {  
    ami            = "ami-0edab43b6fa892279"  
    instance_type  = "t2.micro"  
    key_name       = "ws"  
    vpc_security_group_ids = ["sg-8064fdee"]  
}
```

> _

```
$ terraform plan
```

Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

```
aws_instance.webserver-2: Refreshing state... [id=i-0d7c0088069819ff8]
```

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed

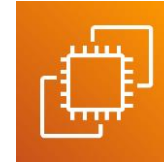


{K}ODE{K}LOUD

Terraform Modules

main.tf

```
resource "aws_instance" "weberver" {  
  # configuration here  
}  
  
resource "aws_key_pair" "key" {  
  # configuration here  
}  
  
resource "aws_security_group" "ssh-access" {  
  # configuration here  
}  
  
resource "aws_s3_bucket" "data-bucket" {  
  # configuration here  
}  
  
resource "aws_dynamodb_table" "user-data" {  
  # configuration here  
}  
  
resource "aws_instance" "web-server-2" {  
  # configuration here  
}
```



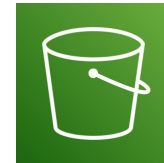
aws_instance



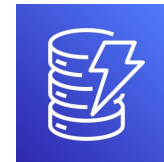
aws_key_pair



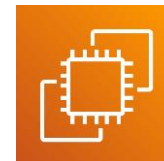
aws_iam_policy



aws_s3_bucket



aws_dynamodb_table



aws_instance

main.tf

```
resource "aws_instance" "webserver" {  
  # configuration here  
}
```

key_pair.tf

```
resource "aws_key_pair" "web" {  
  # configuration here  
}
```

dynamodb_table.tf

```
resource "aws_dynamodb_table" "state-locking" {  
  # configuration here  
}
```

security_group.tf

```
resource "aws_security_group" "ssh-access" {  
  # configuration here  
}
```

ec2_instance.tf

```
resource "aws_instance" "webserver-2" {  
  # configuration here  
}
```

s3_bucket.tf

```
resource "aws_s3_bucket" "terraform-state" {  
  # configuration here  
}
```

> _

\$ ls

```
provider.tf
id_rsa
id_rsa.pub
main.tf
pub_ip.txt
terraform.tfstate.backup
terraform.tfstate
iam_roles.tf
iam_users.tf
security_groups.tf
variables.tf
outputs.tf
s3_buckets.tf
dynamo_db.tf
local.tf
```

Complex Configuration
Files

Duplicate Code

Increased Risk

Limits Reusability

Root Module

```
>_
```

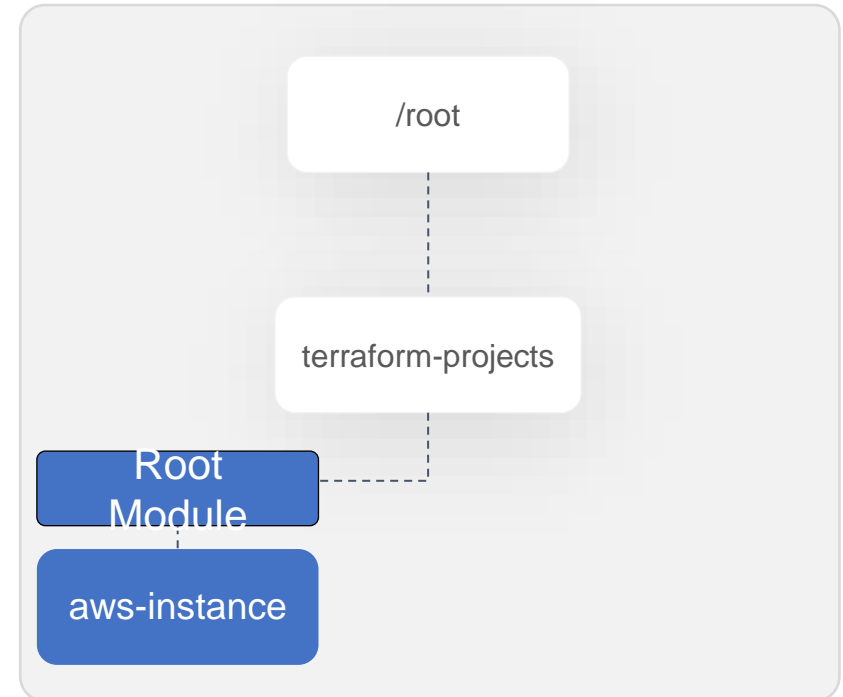
```
$ ls /root/terraform-projects/aws-instance  
main.tf    variables.tf
```

main.tf

```
resource "aws_instance" "webserver" {  
    ami = var.ami  
    instance_type = var.instance_type  
    key_name = var.key  
}
```

variables.tf

```
variable ami {  
    type      = string  
    default   = "ami-0edab43b6fa892279"  
    description = "Ubuntu AMI ID in the ca-  
central-1 region"  
}
```



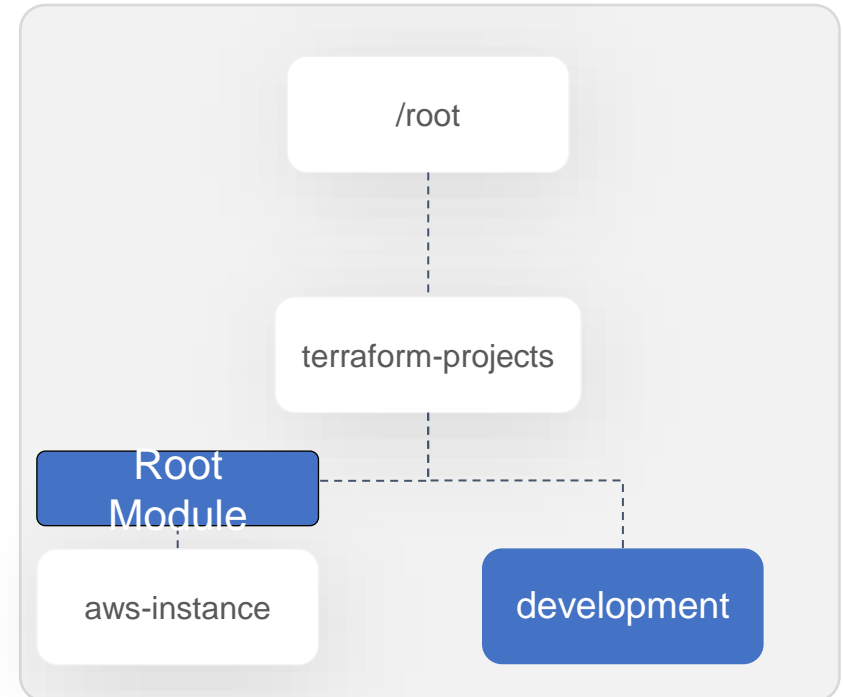
Root Module

```
>_
```

```
$ mkdir /root/terraform-projects/development  
main.tf
```

```
main.tf
```

```
module "dev-webserver" {  
  source = "../aws-instance"  
}
```



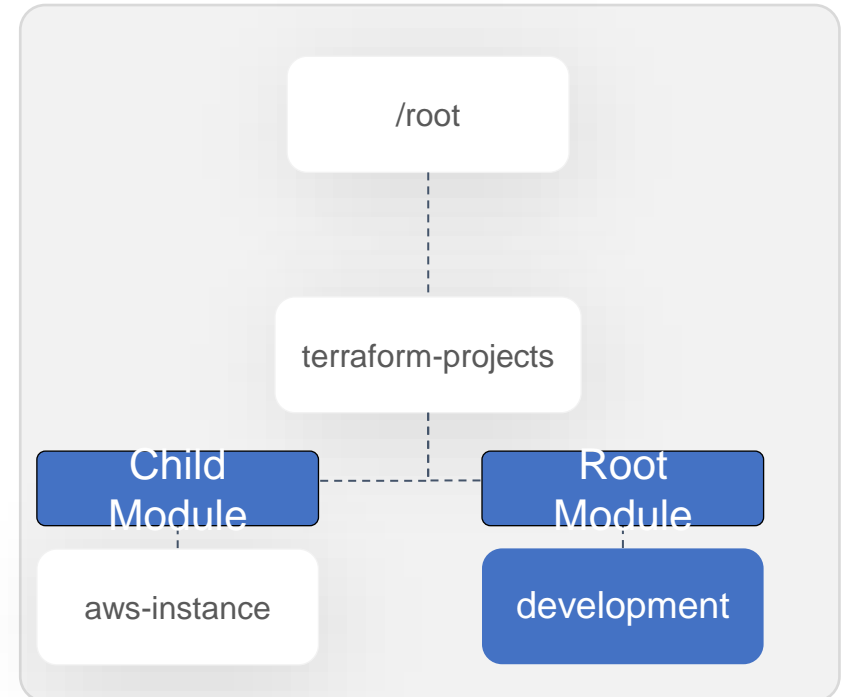
Root Module

```
>_
```

```
$ mkdir /root/terraform-projects/development  
main.tf
```

```
main.tf
```

```
module "dev-webserver" {  
  source = "../aws-instance"  
}
```



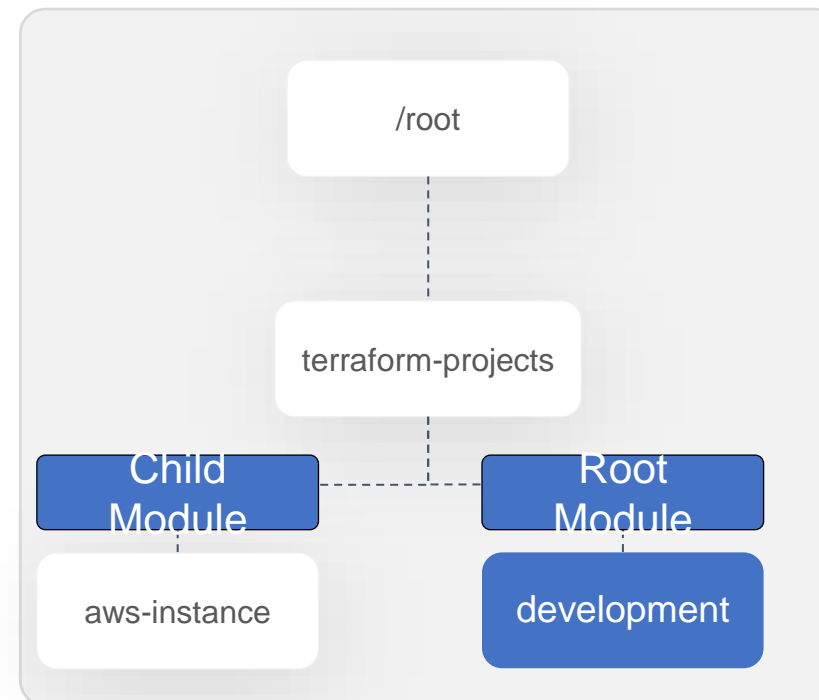
```
> _
```

```
$ mkdir /root/terraform-projects/development
```

```
main.tf
```

```
main.tf
```

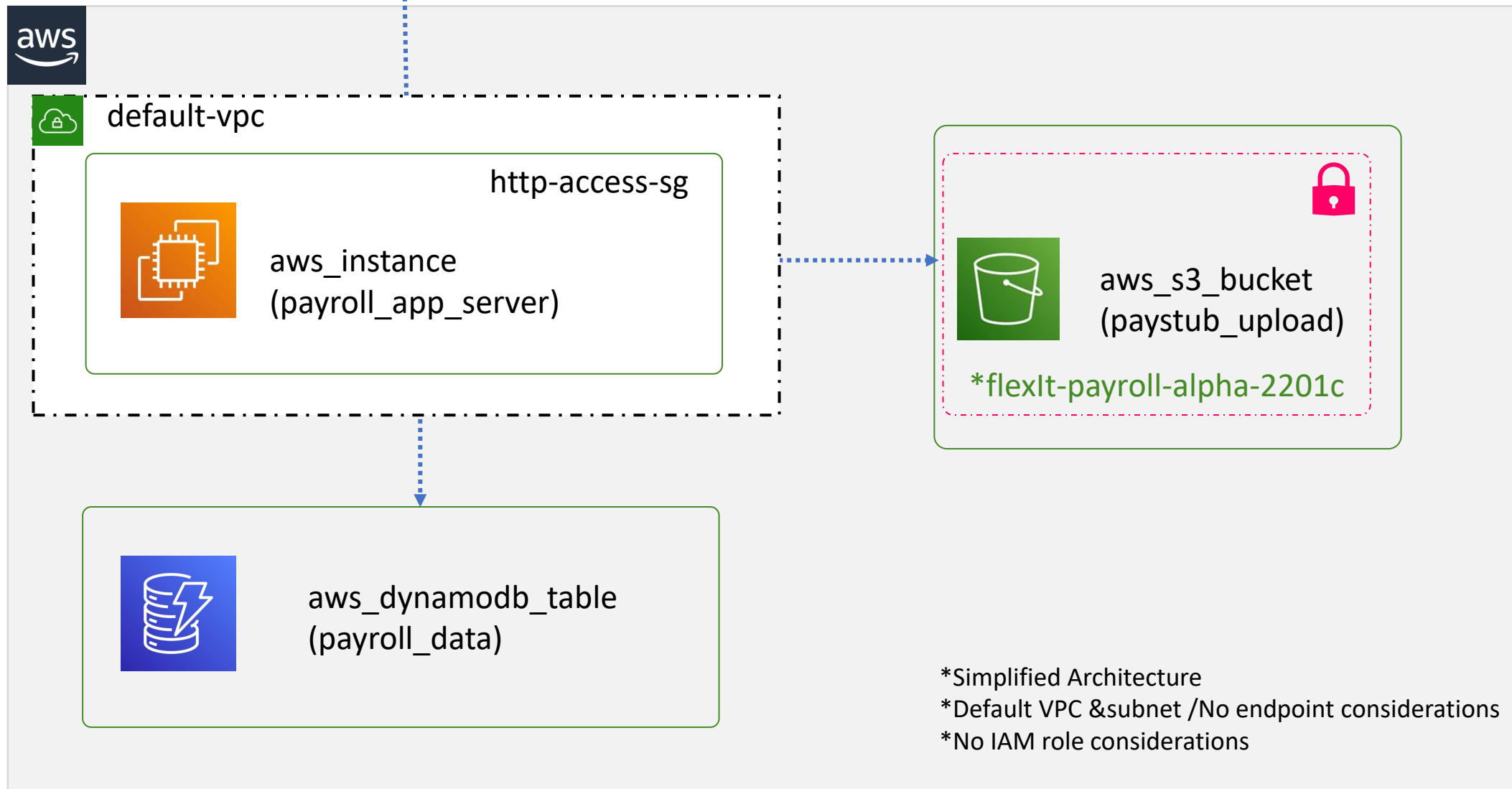
```
module "dev-webserver" {  
  source = "../aws-instance"  
}
```





{K}ODE{K}LOUD

Creating and Using a Module



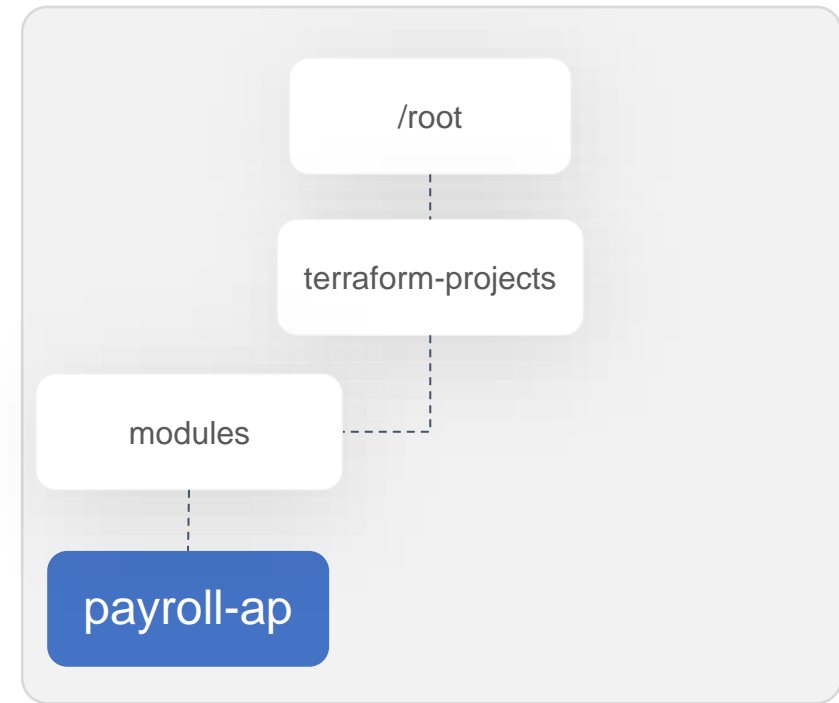
```
>_  
  
$ mkdir /root/terraform-projects/modules/payroll-app  
app_server.tf dynamodb_table.tf s3_bucket.tf variables.tf
```

app_server.tf

```
resource "aws_instance" "app_server" {  
  ami           = var.ami  
  instance_type = "t2.medium"  
  tags = {  
    Name = "${var.app_region}-app-server"  
  }  
  depends_on = [ aws_dynamodb_table.payroll_db,  
                 aws_s3_bucket.payroll_data  
               ]  
}
```

s3_bucket.tf

```
resource "aws_s3_bucket" "payroll_data" {  
  bucket = "${var.app_region}-${var.bucket}"  
}
```

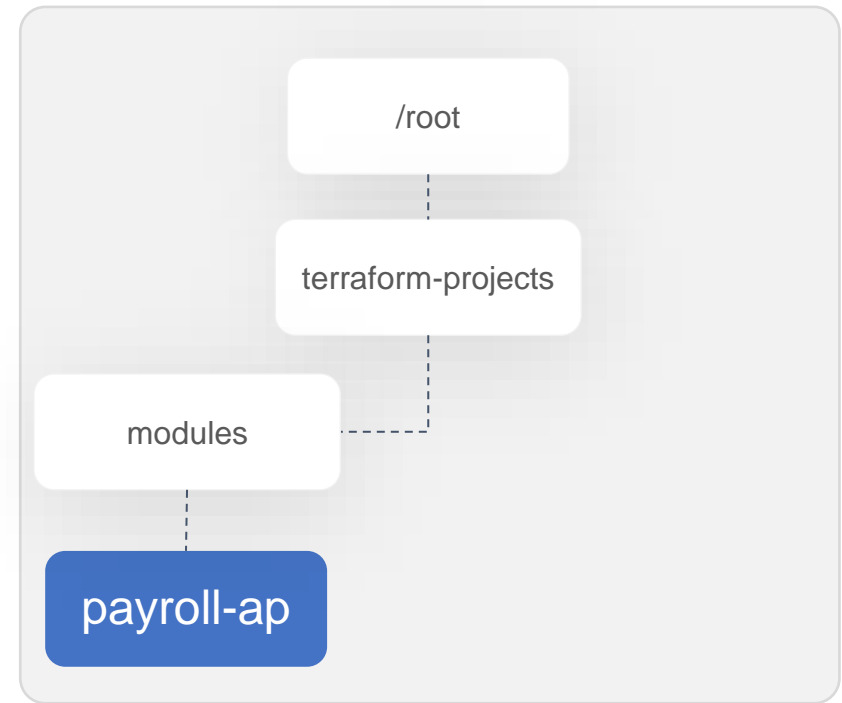


dynamodb_table.tf

```
resource "aws_dynamodb_table" "payroll_db" {  
  name           = "user_data"  
  billing_mode   = "PAY_PER_REQUEST"  
  hash_key      = "EmployeeID"  
  attribute {  
    name = "EmployeeID"  
    type = "N"  
  }  
}
```

variables.tf

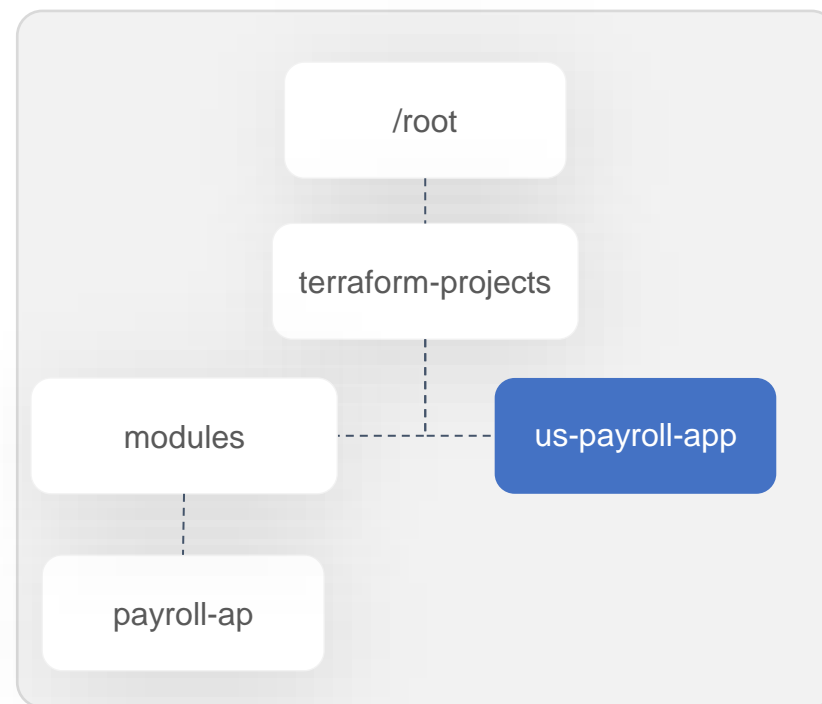
```
variable "app_region" {  
    type = string  
}  
variable "bucket" {  
    default = "flexit-payroll-alpha-22001c"  
}  
variable "ami" {  
    type = string  
}
```



```
> _  
  
$ mkdir /root/terraform-projects/us-payroll-app  
main.tf provider.tf
```

main.tf

```
module "us_payroll" {  
  source = "../modules/payroll-app"  
  app_region = "us-east-1"  
  ami       = "ami-24e140119877avm"  
}
```



> _

```
$ terraform init
```

```
Initializing modules...
```

```
- us_payroll in .terraform/modules/us_payroll
```

```
Initializing the backend...
```

```
Initializing provider plugins...
```

- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.11.0...
- Installed hashicorp/aws v3.11.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking changes, we recommend adding version constraints in a `required_providers` block in your configuration, with the constraint strings suggested below.

```
* hashicorp/aws: version = "~> 3.11.0"
```

```
Terraform has been successfully initialized!
```

>_

```
$ terraform apply
```

```
.  
.
```

Terraform will perform the following actions:

```
# module.us_payroll.aws_dynamodb_table.payroll_db will be created  
+ resource "aws_dynamodb_table" "payroll_db" {  
  + arn                = (known after apply)  
  + billing_mode       = "PAY_PER_REQUEST"  
  + hash_key           = "EmployeeID"  
  + name               = "user_data"  
.  
.  
# module.us_payroll.aws_instance.app_server will be created  
+ resource "aws_instance" "app_server" {  
  + ami                = "ami-24e140119877avm"  
  + instance_type      = "t2.medium"  
.  
.  
+ resource "aws_s3_bucket" "payroll_data" {  
  + acceleration_status = (known after apply)  
  + acl                 = "private"  
  + arn                 = (known after apply)  
  + bucket              = "us-east-1-flexit-payroll-alpha-22001c"  
}
```

Enter a value: yes

module.us_payroll.aws_dynamodb_table.payroll_db: Creating...

module.us_payroll.aws_s3_bucket.payroll_data: Creating...

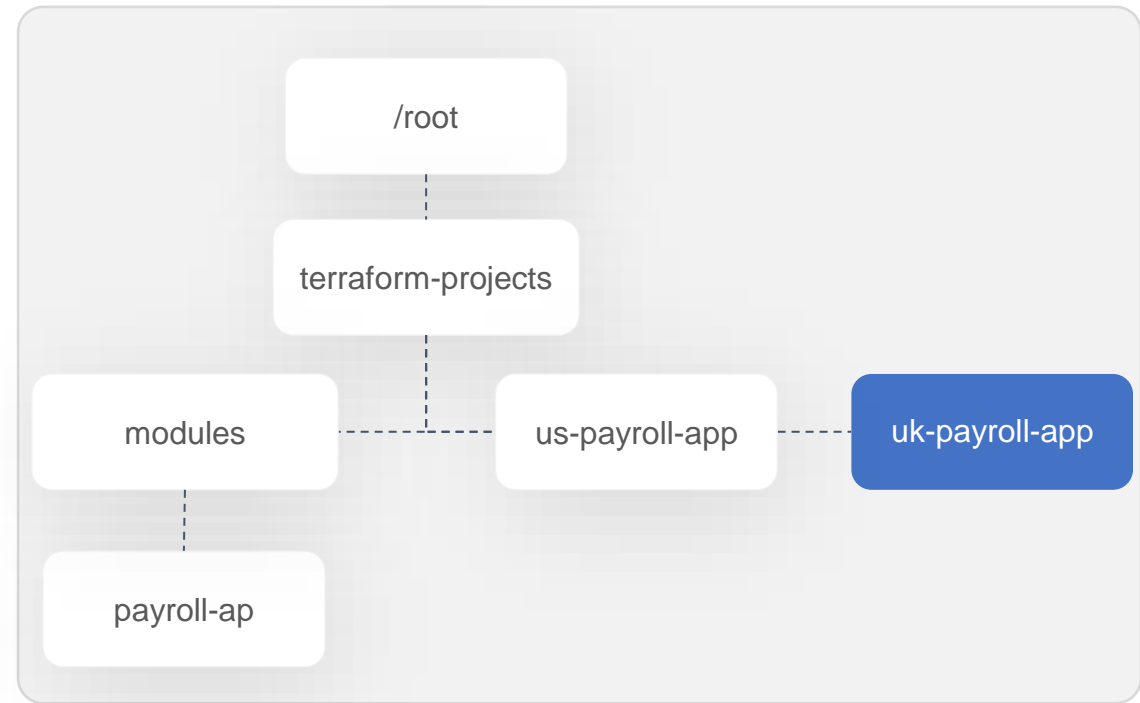
```
> _  
  
$ mkdir /root/terraform-projects/uk-payroll-app  
main.tf provider.tf
```

main.tf

```
module "uk_payroll" {  
  source = "../modules/payroll-app"  
  app_region = "eu-west-2"  
  ami       = "ami-35e140119877avm"  
}
```

provider.tf

```
provider "aws" {  
  region = "eu-west-2"  
}
```



>_

```
$ terraform apply
```

```
.  
.
```

Terraform will perform the following actions:

```
# module.us_payroll.aws_dynamodb_table.payroll_db will be created  
+ resource "aws_dynamodb_table" "payroll_db" {  
  + arn              = (known after apply)  
  + billing_mode     = "PAY_PER_REQUEST"  
  + hash_key        = "EmployeeID"  
  + name            = "user_data"  
.  
.  
# module.us_payroll.aws_instance.app_server will be created  
+ resource "aws_instance" "app_server" {  
  + ami              = "ami-35e140119877avm"  
  + instance_type    = "t2.medium"  
.  
.  
+ resource "aws_s3_bucket" "payroll_data" {  
  + acceleration_status = (known after apply)  
  + acl                 = "private"  
  + arn                 = (known after apply)  
  + bucket              = "eu-west-2-flexit-payroll-alpha-22001c"  
}  
Enter a value: yes
```

```
module.us_payroll.aws_dynamodb_table.payroll_db: Creating...
```

```
module.us_payroll.aws_s3_bucket.payroll_data: Creating...
```

```
module.us_payroll.aws_dynamodb_table.payroll_db: Creation complete after 1s [id=user_data]
```

-
-

Terraform will perform the following actions:

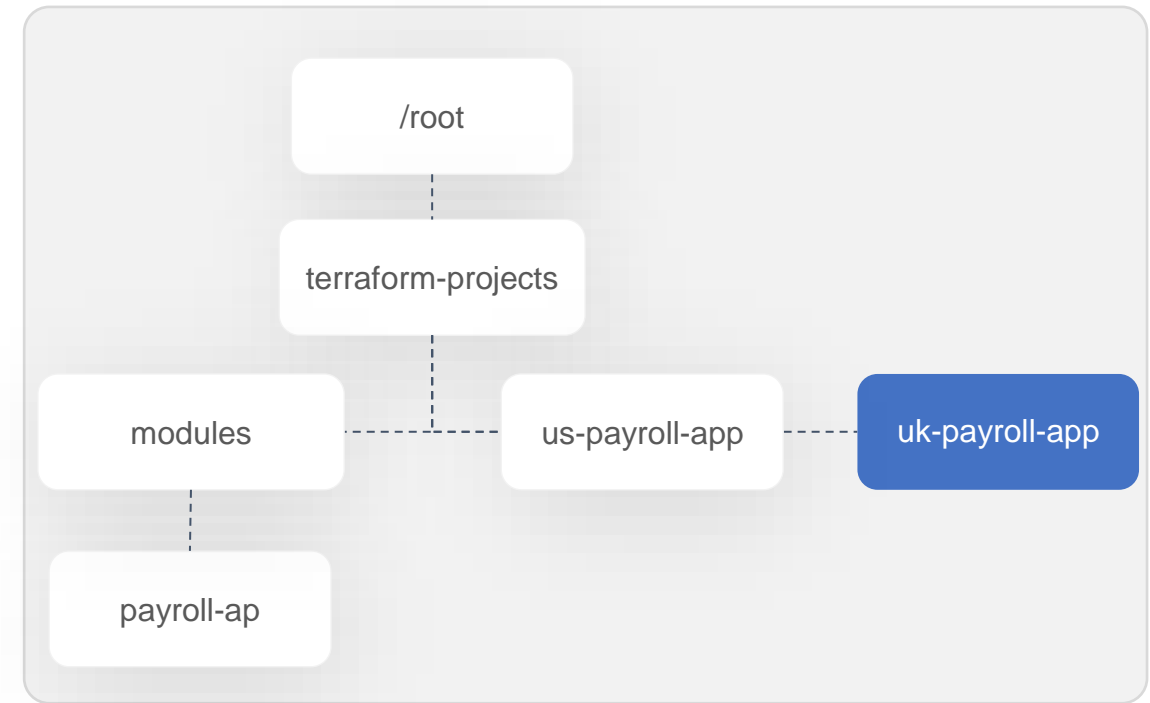
```
# module.us_payroll.aws_dynamodb_table.payroll_db will be created
+ resource "aws_dynamodb_table" "payroll_db" {
+   arn                = (known after apply)
+   billing_mode        = "PAY_PER_REQUEST"
+   hash_key            = "EmployeeID"
+   name                = "user_data"
```

-
-

```
# module.us_payroll.aws_instance.app_server will be created
+ resource "aws_instance" "app_server" {
+   ami                = "ami-35e140119877avm"
```

main.tf

```
module "us_payroll" {  
  source = "../modules/payroll-app"  
  app_region = "eu-west-2"  
  ami      = "ami-35e140119877avm"  
}
```



Simpler Configuration Files

Lower Risk

Re-Usability

Standardized Configuration



{K}ODE{K}LOUD

Using Modules from Registry

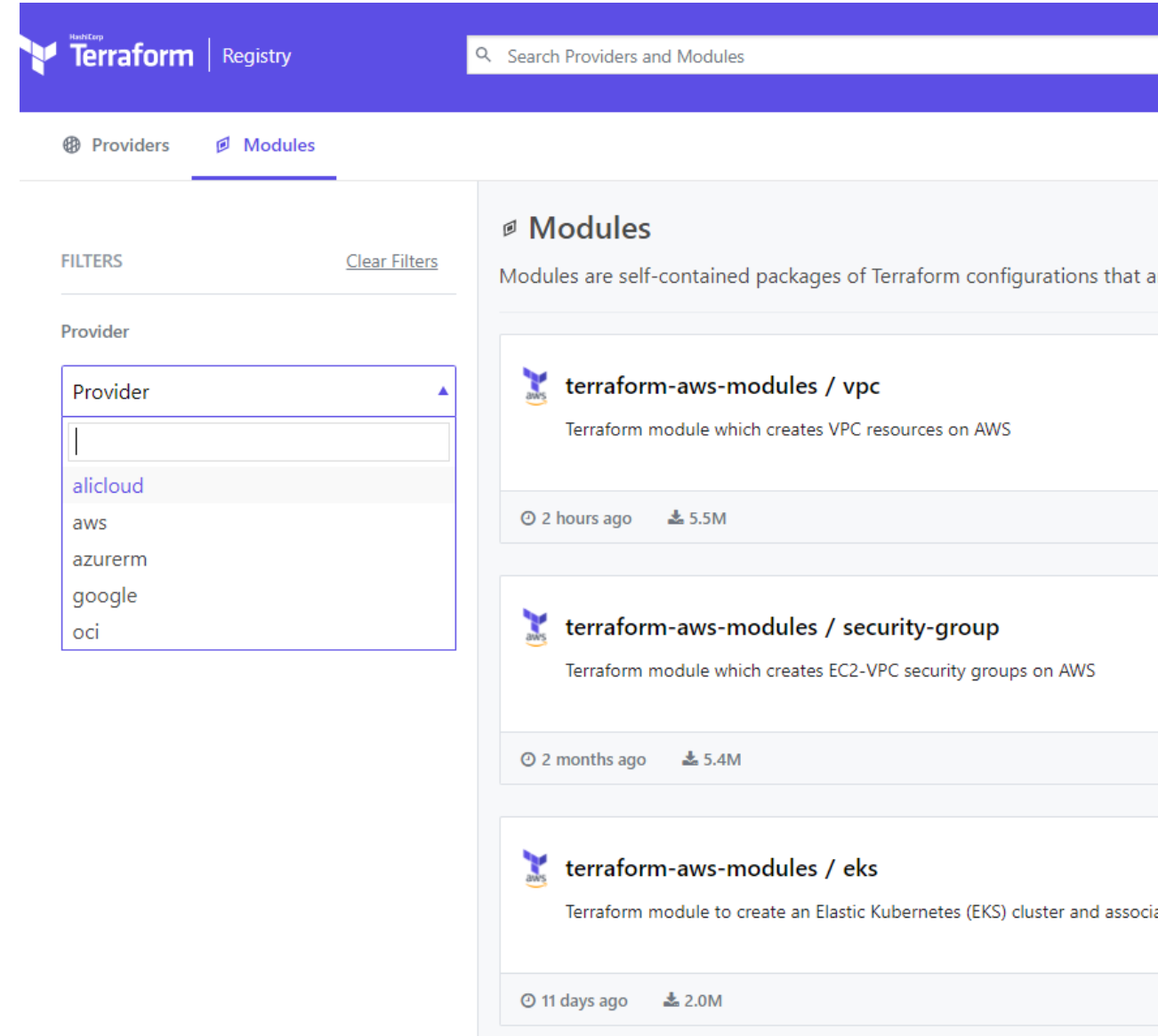
Local Module

```
main.tf

module "dev-webserver" {

    source = "../aws-instance/"

    key    = "webserver"
}
```




The screenshot shows the Terraform Registry interface. At the top, there's a header with the Terraform logo and a search bar labeled "Search Providers and Modules". Below the header, there are tabs for "Providers" and "Modules", with "Modules" being the active tab. On the left, there's a "FILTERS" section with a "Clear Filters" link. Below this, a "Provider" dropdown menu is open, showing a list of providers: "alicloud", "aws", "azurerm", "google", and "oci". The "aws" provider is selected. On the right, there's a "Modules" section with a description: "Modules are self-contained packages of Terraform configurations that a...". Below this, there are three module cards, each with the Terraform logo, the module name, a description, and download statistics. The first card is for "terraform-aws-modules / vpc", described as "Terraform module which creates VPC resources on AWS", with a download count of 5.5M and a timestamp of "2 hours ago". The second card is for "terraform-aws-modules / security-group", described as "Terraform module which creates EC2-VPC security groups on AWS", with a download count of 5.4M and a timestamp of "2 months ago". The third card is for "terraform-aws-modules / eks", described as "Terraform module to create an Elastic Kubernetes (EKS) cluster and associ...", with a download count of 2.0M and a timestamp of "11 days ago".

Modules


Modules are self-contained packages of Terraform configurations that a...

- terraform-aws-modules / vpc**
Terraform module which creates VPC resources on AWS
2 hours ago 5.5M
- terraform-aws-modules / security-group**
Terraform module which creates EC2-VPC security groups on AWS
2 months ago 5.4M
- terraform-aws-modules / eks**
Terraform module to create an Elastic Kubernetes (EKS) cluster and associ...
11 days ago 2.0M


Terraform Registry




terraform-aws-modules/security-group
Terraform module which creates EC2-VPC security groups on AWS




dcos-terraform/security-groups
Create DC/OS related security groups




Azure/network-security-group
Terraform module to create a network security group and assign it to the specified subnet




devops-workflow/security-group
Terraform module which creates EC2-VPC security groups on AWS



claranet/nsg
Terraform module for Azure Network Security Group



security-group 
AWS


Terraform module which creates EC2-VPC security groups on AWS


Published August 20, 2020 by [terraform-aws-modules](#)

Module managed by [antonbabenko](#)

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an issue)

 Submodules ▾

 Examples ▾

Terraform Module



security-group

AWS

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by [terraform-aws-modules](#)

Module managed by [antonbabenko](#)

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group ([report an issue](#))

Submodules ▼

Examples ▼

Version 3.16.0 (latest) ▼

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {  
  source = "terraform-aws-modules/security-group,  
  version = "3.16.0"  
  # insert the 2 required variables here  
}
```


Terraform Module



security-group

AWS

Version 3.16.0 (latest) ▾

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by [terraform-aws-modules](#)

Module managed by [antonbabenko](#)

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group ([report an issue](#))

Submodules ▾

Examples ▾

- activemq
- alertmanager
- carbon-relay-ng
- cassandra
- consul
- docker-swarm
- elasticsearch
- grafana
- graphite-statsd
- http-80
- http-8080
- https-443

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {  
  source = "terraform-aws-modules/security-group,  
  version = "3.16.0"  
  # insert the 2 required variables here  
}
```

main.tf

```
module "security-group_ssh" {  
  source = "terraform-aws-modules/security-group/aws/modules/ssh"  
  version = "3.16.0"  
  # insert the 2 required variables here  
  vpc_id = "vpc-7d8d215"  
  ingress_cidr_blocks = [ "10.10.0.0/16" ]  
  name = "ssh-access"  
}
```

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {  
  source = "terraform-aws-modules/security-group/  
  version = "3.16.0"  
  # insert the 2 required variables here  
}
```

> _

```
$ terraform get
```

```
Downloading terraform-aws-modules/security-group/aws 3.16.0 for security-group_ssh...  
- security-group_ssh in .terraform\modules\security-group_ssh\modules\ssh
```



Terraform Functions

Functions

main.tf

```
resource "aws_iam_policy" "adminUser" {
  name     = "AdminUsers"
  policy   = file("admin-policy.json")
}

resource "local_file" "pet" {
  filename = var.filename
  count    = length(var.filename)
}
```

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  for_each = toset(var.region)
}

variable region {
  type        = list
  default     = ["us-east-1",
                "us-east-1",
                "ca-central-1"]
  description = "A list of AWS Regions"
}
```

>_

```
$ terraform console
> file("/root/terraform-projects/main.tf")
resource "aws_instance" "development" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}
> length(var.region)
3
> toset(var.region)
[
  "ca-central-1",
  "us-east-1",
]
>
```

Functions

Numeric Functions

String Functions

Collection Functions

Type Conversion
Functions

Numeric Functions

variables.tf

```
variable "num" {  
  type = set(number)  
  default = [ 250, 10, 11, 5]  
  description = "A set of numbers"  
}
```

> _

\$ terraform console

```
> max (-1, 2, -10, 200, -250)  
200
```

```
> min (-1, 2, -10, 200, -250)  
-250
```

```
> max(var.num...)  
250
```

```
> ceil(10.1)  
11
```

```
> ceil(10.9)  
11
```

```
> floor(10.1)  
10
```

```
> floor(10.9)  
10
```

String Functions

variables.tf

```
variable "ami" {  
  type = string  
  default = "ami-xyz,AMI-ABC,ami-efg"  
  description = "A string containing ami ids"  
}
```

> _

```
$ terraform console  
  
> split(",", "ami-xyz,AMI-ABC,ami-efg")  
[ "ami-xyz", "AMI-ABC", "ami-efg" ]  
  
> split(",", var.ami)  
[ "ami-xyz", "AMI-ABC", "ami-efg" ]  
  
> lower(var.ami)  
ami-xyz,ami-abc,ami-efg  
  
> upper(var.ami)  
AMI-XYZ,AMI-ABC,AMI-EFG  
  
> title(var.ami)  
Ami-XYZ,AMI-ABC,Ami-Efg  
  
> substr(var.ami, 0, 7)  
ami-xyz  
  
> substr(var.ami, 8, 7)  
AMI-ABC  
  
> substr(var.ami, 16, 7)  
ami-efg
```


String Functions

variables.tf

```
variable "ami" {  
  type = list  
  default = ["ami-xyz", "AMI-ABC", "ami-efg"]  
  description = "A list of numbers"  
}
```

>_

```
$ terraform console  
  
> join(",", ["ami-xyz", "AMI-ABC", "ami-efg"])  
ami-xyz,AMI-ABC,ami-efg  
  
> join(",", var.ami)  
ami-xyz,AMI-ABC,ami-efg
```

Collection Functions

variables.tf

```
variable "ami" {  
  type = list  
  default = ["ami-xyz", "AMI-ABC", "ami-efg"]  
  description = "A list of numbers"  
}
```

>_

```
$ terraform console  
> length(var.ami)  
3  
  
> index(var.ami, "AMI-ABC")  
1  
  
> element(var.ami,2)  
ami-efg  
  
> contains(var.ami, "AMI-ABC")  
true  
  
> contains(var.ami, "AMI-XYZ")  
false
```

Map Functions

variables.tf

```
variable "ami" {  
  type = map  
  default = { "us-east-1" = "ami-xyz",  
              "ca-central-1" = "ami-efg",  
              "ap-south-1" = "ami-ABC"  
            }  
  description = "A map of AMI ID's for specific regions"  
}
```

>_

```
$ terraform console  
> keys(var.ami)  
[  
  "ap-south-1",  
  "ca-central-1",  
  "us-east-1",  
]  
  
> values(var.ami)  
[  
  "ami-ABC",  
  "ami-efg",  
  "ami-xyz",  
]  
  
> lookup(var.ami, "ca-central-1")  
ami-efg
```

Map Functions

variables.tf

```
variable "ami" {  
  type = map  
  default = { "us-east-1" = "ami-xyz",  
              "ca-central-1" = "ami-efg",  
              "ap-south-1" = "ami-ABC"  
            }  
  description = "A map of AMI ID's for specific regions"  
}
```

>_

```
$ terraform console
```

```
> lookup(var.ami, "us-west-2")
```

```
Error: Error in function call
```

```
on <console-input> line 1:
```

```
(source code not available)
```

```
|-----
```

```
| var.ami is map of string with 3 elements
```

```
Call to function "lookup" failed: lookup failed  
to find 'us-west-2'.
```

```
> lookup (var.ami, "us-west-2", "ami-pqr")
```

```
ami-pqr
```



{K}ODE{K}LOUD

Operators & Conditional Expressions

Numeric Operators

> _

\$ terraform console

> 1 + 2

3

> 5 - 3

2

> 2 * 2

4

> 8 / 2

4

Equality Operators

```
> _
```

```
$ terraform console
```

```
> 8 == 8  
true
```

```
8 == 7  
false
```

```
> 8 != "8"  
true
```


Comparison Operators

```
>_  
$ terraform console
```

```
> 5 > 7  
false
```

```
> 5 > 4  
true
```

```
> 5 > 5  
False
```

```
> 5 >= 5  
true
```

```
> 4 < 5  
true
```

```
> 3 <= 4  
true
```

Logical Operators

> _

```
$ terraform console
```

```
> 8 > 7 && 8 < 10  
true
```

```
> 8 > 10 && 8 < 10  
false
```

```
> 8 > 9 || 8 < 10  
True
```

```
> var.special  
true
```

```
> ! var.special  
false
```

```
> ! (var.b > 30)  
true
```



variables.tf

```
variable special {  
  type      = bool  
  default   = true  
  description = "Set to true to  
                use special characters"  
}  
  
variable b {  
  type = number  
  default = 25  
}
```

Logical Operators

> _

```
$ terraform console
```

```
> var.a > var.b  
true
```

```
> var.a < var.b  
false
```

```
> var.a + var.b  
75
```



variables.tf

```
variable a {  
    type = number  
    default = 50  
}  
variable b {  
    type = number  
    default = 25  
}
```

main.tf

```
resource "random_password" "password-generator" {
  length = var.length
}

output password {
  value = random_password.password-generator.result
}
```

variables.tf

```
variable length {
  type        = number
  description = "The length of the password"
}
```

>_

```
$ terraform apply -var=length=5 -auto-approve
random_password.password-generator: Creating...
random_password.password-generator: Creation
complete after 0s [id=none]
```

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

```
password = sjsrW]
```

```
$ if [ $length -lt 8 ]
then
    length=8;
    echo $length;
else
    echo $length;
fi
# Generate Password
```

Condition

If True

If False

condition

If True

If False

main.tf

```
resource "random_password" "password-generator" {  
  length = {var.length < 8 ? 8 : var.length}  
}  
  
output password {  
  value = random_password.password-generator.result  
}
```

condition ? true_val : false_val

variables.tf

```
variable length {  
  type        = number  
  description = "The length of the password"  
}
```

```
$ if [ $length -lt 8 ]  
then  
    length=8;  
    echo $length;  
else  
    echo $length;  
fi  
# Generate Password
```

Condition

If True

If False

```
>_
```

```
$ terraform apply -var=length=5
```

Terraform will perform the following actions:

```
# random_password.password-generator will be created
+ resource "random_password" "password-generator" {
  + id          = (known after apply)
  + length      = 8
}
```

.

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

```
password = &(1Beiaq
```

```
$ terraform apply -var=length=12
```

Terraform will perform the following actions:

```
# random_password.password-generator must be replaced
-/+ resource "random_password" "password-generator" {
  ~ id          = "none" -> (known after apply)
  ~ length      = 8 -> 12 # forces replacement.
}
```

.

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

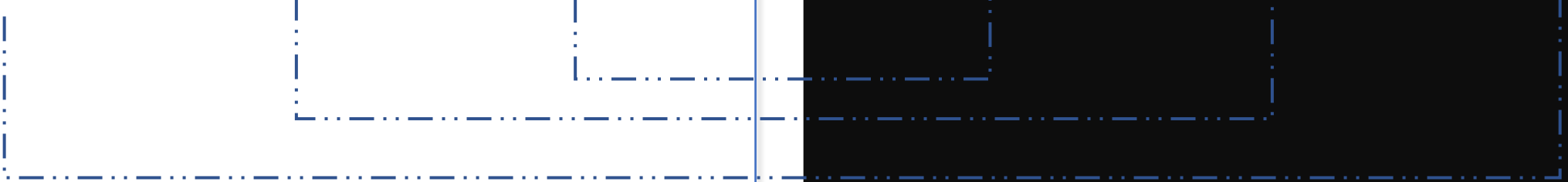
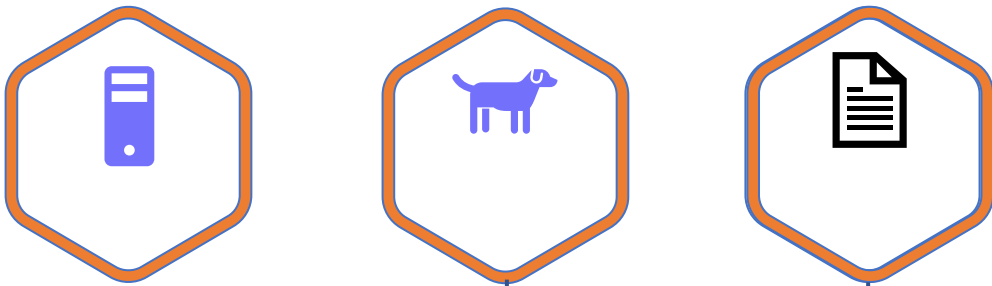
```
password = 8B@o}{cUzrZ7
```



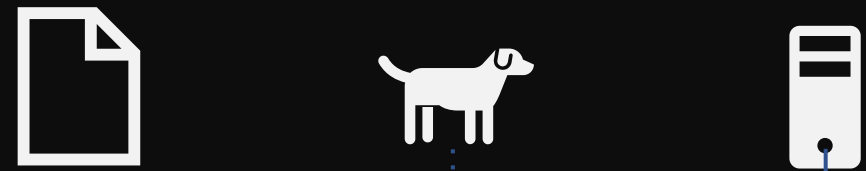
The background of the slide features a dark blue field with several concentric, semi-transparent hexagonal shapes in a lighter shade of blue. These hexagons are centered around the text, creating a layered, geometric effect. The text is white and positioned in the center of the composition.

Terraform Workspaces

Real World Infrastructure



terraform.tfstate





variables.tf
main.tf
terraform.tfstate

/root/terraform-projects/projectA



variables.tf
main.tf
terraform.tfstate

/root/terraform-projects/projectB

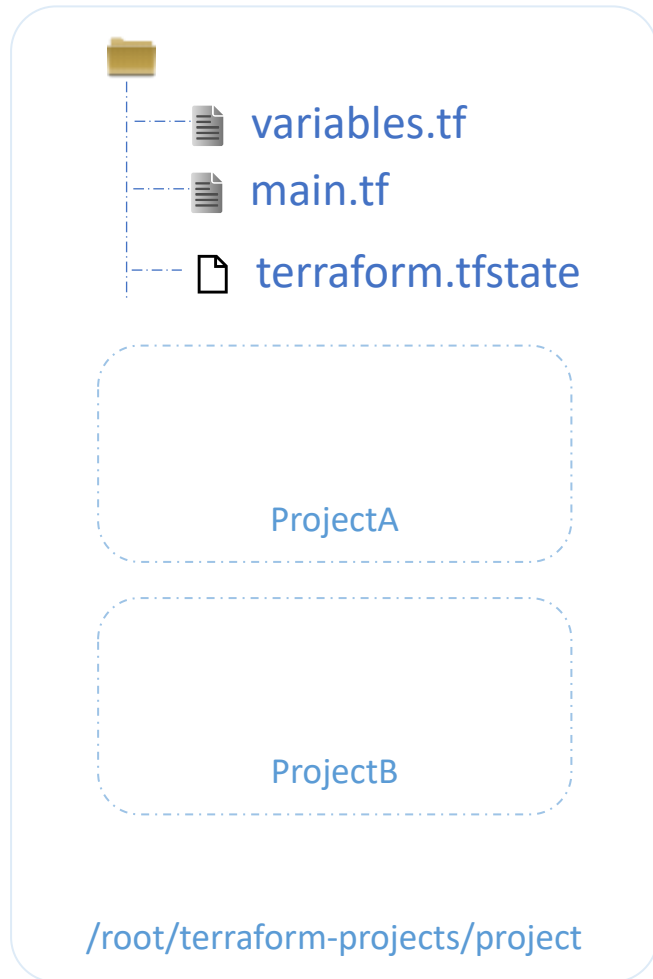
main.tf

```
resource "aws_instance" "projectA" {  
  ami = "ami-0edab43b6fa892279"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "ProjectA"  
  }  
}
```

main.tf

```
resource "aws_instance" "projectB" {  
  ami = "ami-0c2f25c1f66a1ff4d"  
  instance_type = "t2.micro"  
  tags = {  
    Name = "ProjectB"  
  }  
}
```

Workspace



```
main.tf

resource "aws_instance" "projectA" {
  ami = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name = "ProjectA"
  }
}
```

Workspace

>_

```
$ terraform workspace new ProjectA
```

```
Created and switched to workspace "ProjectA"!
```

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

```
$ terraform workspace list
```

```
default
```

```
* ProjectA
```



Region: ca-central-1

AMI: `ami-0edab43b6fa892279`

Instance Type: t2.micro

ProjectA

Region: ca-central-1

AMI: `ami-0c2f25c1f66a1ff4d`

Instance Type: t2.micro

ProjectB

`/root/terraform-projects/project`

variables.tf

```
variable region {
  default = "ca-central-1"
}
variable instance_type {
  default = "t2.micro"
}
variable ami {
  type      = map
  default   = {
    "ProjectA" = "ami-0edab43b6fa892279",
    "ProjectB" = "ami-0c2f25c1f66a1ff4d"
  }
}
```

main.tf

```
resource "aws_instance" "projectA" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name = "ProjectA"
  }
}
```



> _

```
$ terraform console
> terraform.workspace
ProjectA

> lookup(var.ami, terraform.workspace)
ami-0edab43b6fa892279
```

Instance Type: t2.micro

ProjectB

/root/terraform-projects/project

variables.tf

```
variable region {
  default = "ca-central-1"
}
variable instance_type {
  default = "t2.micro"
}
variable ami {
  type      = map
  default   = {
    "ProjectA" = "ami-0edab43b6fa892279",
    "ProjectB" = "ami-0c2f25c1f66a1ff4d"
  }
}
```

main.tf

```
resource "aws_instance" "projectA" {
  ami = lookup(var.ami, terraform.workspace)
  instance_type = var.instance_type
  tags = {
    Name = terraform.workspace
  }
}
```

>_

```
$ terraform plan
```

Terraform will perform the following actions:

```
# aws_instance.project will be created
+ resource "aws_instance" "project" {
  + ami                = "ami-0edab43b6fa892279"
  + instance_type      = "t2.micro"
  + tags               = {
    + "Name" = "ProjectA"
  }
.
.
```

```
$ terraform workspace new ProjectB
```

Created and switched to workspace "ProjectB"!

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

>_

```
$ terraform plan
```

Terraform will perform the following actions:

```
# aws_instance.project will be created
+ resource "aws_instance" "project" {
  + ami                = "ami-0c2f25c1f66a1ff4d"
  + instance_type      = "t2.micro"
  + tags               = {
    + "Name" = "ProjectB"
  }
.
.
.
```

```
$ terraform workspace select ProjectA
```

Switched to workspace "ProjectA".

>_

```
$ ls
```

```
main.tf  provider.tf  terraform.tfstate.d  variables.tf
```

```
$ tree terraform.tfstate.d/
```

```
terraform.tfstate.d/
```

```
|-- ProjectA
```

```
|   |-- terraform.tfstate
```

```
`-- ProjectB
```

```
    |-- terraform.tfstate
```

```
2 directories, 2 files
```